

Intro to Time Series

Archived study guides

These archived materials are no longer updated. This archive was exported from a saved copy of these guides.

Export date: 2026-10-05T19:26:30.244Z

Contents

- Unit 1 – Introduction to Time Series - 4 guides
- Unit 2 – Time Series Components - 4 guides
- Unit 3 – Stationary vs Non-Stationary Time Series - 4 guides
- Unit 4 – Autocorrelation and Partial Autocorrelation - 3 guides
- Unit 5 – Moving Average & Exponential Smoothing - 4 guides
- Unit 6 – ARIMA Models - 4 guides
- Unit 7 – Model Selection & Diagnostic Checking - 3 guides
- Unit 8 – Forecasting with Time Series Models - 4 guides
- Unit 9 – Seasonal ARIMA Models - 3 guides
- Unit 10 – Multivariate Time Series Analysis - 3 guides
- Unit 11 – Spectral Analysis in Time Series - 3 guides
- Unit 12 – State-Space Models & Kalman Filtering - 3 guides
- Unit 13 – Time Series Regression & Intervention Analysis - 3 guides
- Unit 14 – Volatility Models: ARCH and GARCH - 3 guides
- Unit 15 – Time Series in Finance and Economics - 3 guides
- Unit 16 – Environmental Science Applications - 3 guides

Unit 1 – Introduction to Time Series

1.1 Definition and characteristics of time series data

Time series data is a sequence of observations recorded at regular intervals, like hourly stock prices or daily temperature readings. It's characterized by temporal dependence, where current values are influenced by past ones, and often includes components like trends, seasonality, and cyclical patterns.

Understanding time series data is crucial for forecasting and uncovering underlying patterns in various fields. From finance to environmental studies, this type of data helps us analyze how variables change over time, making it a powerful tool for decision-making and trend analysis.

Introduction to Time Series Data

Definition of time series data

- Sequence of observations recorded at regular time intervals (hourly, daily, monthly)
- Each observation associated with a specific timestamp or date
- Key components include trend, seasonality, cyclical component, and irregularity or noise
- Trend represents long-term increase or decrease over time
- Seasonality refers to recurring patterns at fixed intervals (holidays, summer months)
- Cyclical component captures patterns over longer periods without fixed frequency (business cycles)
- Irregularity or noise encompasses random fluctuations not explained by other components

Characteristics of time series data

- Temporal dependence whereby current observations influenced by previous values
- Crucial for forecasting and understanding underlying patterns
- Seasonality exhibits regular, predictable patterns recurring over fixed time intervals
- Increased retail sales during holiday seasons
- Higher electricity consumption in summer months
- Autocorrelation measures correlation between a variable's current and past values
- Positive autocorrelation indicates high values followed by high values, low by low
- Negative autocorrelation suggests high values likely followed by low values, and vice versa
- Stationarity assumes statistical properties remain constant over time
- Non-stationary series may require transformations (differencing) to achieve stationarity
- Trend-cycle component represents overall long-term pattern
- Trend refers to general direction (increasing or decreasing)
- Cycle captures longer-term fluctuations around the trend

- Sampling frequency determines rate at which observations are recorded
- High-frequency data collected at short intervals (hourly, daily)
- Low-frequency data recorded at longer intervals (monthly, quarterly, annually)

Time series vs cross-sectional data

- Time series data consists of observations recorded over time for a single entity
- Focuses on evolution of variables over time
- Daily stock prices of a particular company over a year
- Cross-sectional data comprises observations collected at a single point in time across multiple entities
- Focuses on relationships between variables at a specific moment
- Income levels of individuals in a city surveyed on a specific date

Examples of time series data

- Finance and economics
- Stock prices, exchange rates, GDP, inflation rates
- Environmental studies
- Temperature measurements, air quality indices, sea level records
- Healthcare
- Disease incidence rates, hospital admissions, patient vital signs
- Energy
- Electricity consumption, oil prices, renewable energy production
- Social media and web analytics
- User engagement metrics, website traffic, social media post interactions
- Meteorology
- Weather variables (temperature, humidity, wind speed, precipitation)
- Epidemiology
- Disease case counts, mortality rates, vaccination rates
- Transportation
- Traffic volume, public transit ridership, flight passenger counts
- Retail and e-commerce
- Sales figures, customer transactions, inventory levels
- Sensor data
- Readings from IoT devices (smart meters, wearables, industrial sensors)

1.2 Applications and importance of time series analysis

Time series analysis is a powerful tool for understanding data that changes over time. It's used in economics, finance, and environmental studies to spot trends, make predictions, and guide decisions based on historical patterns.

By analyzing time-dependent data, organizations can forecast future values, identify anomalies, and uncover relationships between variables. This helps with planning, risk management, and making informed choices across various fields and industries.

Applications and Importance of Time Series Analysis

Importance of time series analysis

- Crucial for understanding and making decisions based on data that changes over time
- Many real-world phenomena generate time-dependent data
- Economic indicators (GDP, inflation, unemployment rates)
- Financial markets (stock prices, exchange rates)
- Environmental variables (temperature, precipitation, air quality)
- Social media metrics (user engagement, sentiment analysis)
- Enables organizations and individuals to:
 - Identify trends, patterns, and relationships in data
 - Make data-driven decisions
 - Forecast future values and plan accordingly

Applications across diverse fields

- Economics:
 - Analyzes economic indicators to assess the health of an economy
 - Forecasts future economic trends and guides policy decisions
 - Studies the impact of events (recessions, policy changes) on economic variables
- Finance:
 - Analyzes stock prices, exchange rates, and other financial data to inform investment decisions
 - Develops trading strategies based on historical patterns and trends
 - Assesses financial risk and portfolio performance
- Environmental studies:
 - Monitors and analyzes environmental variables (temperature, air quality) over time
 - Identifies trends and patterns related to climate change and other environmental issues
 - Forecasts future environmental conditions and their potential impacts

Role in forecasting and decisions

- Forecasting:
 - Enables the prediction of future values based on historical data
 - Applies forecasting techniques (exponential smoothing, ARIMA models) to various domains
 - Generates accurate forecasts to help organizations plan and allocate resources effectively
- Decision-making:
 - Provides insights into past and current trends to inform decision-making
 - Identifies patterns and relationships in data to guide strategic planning and policy formulation
 - Supports proactive decision-making and risk management through forecasts

Benefits for data interpretation

- Trend identification:
 - Detects long-term trends in data
 - Informs strategic planning and resource allocation based on trend understanding
- Pattern recognition:
 - Reveals recurring patterns (seasonality, cycles) in data
 - Optimizes operations and anticipates future events by identifying patterns
- Relationship discovery:
 - Uncovers relationships between different variables over time
 - Identifies lead-lag relationships and correlations to provide valuable insights for decision-making
- Anomaly detection:
 - Identifies unusual or unexpected events in data
 - Prevents issues, identifies opportunities, and improves overall performance by detecting anomalies

1.3 Basic time series plots and data visualization

Time series plots are essential tools for visualizing and understanding data that changes over time. These plots help identify trends, seasonality, and patterns in data, making them crucial for analyzing everything from stock prices to weather patterns.

Various plot types, including line, scatter, and bar plots, serve different purposes in time series analysis. Interpreting these plots involves recognizing trends, seasonality, cyclic patterns, and outliers. Visualization techniques like smoothing and multiple series comparisons enhance our ability to extract meaningful insights from time series data.

Basic Time Series Plots and Data Visualization

Time series plot creation

- Line plots connect consecutive data points with lines to display the evolution of a time series over time and are suitable for visualizing trends, seasonality (monthly sales), and other patterns
- Scatter plots display individual data points without connecting them and are useful for identifying outliers (unusual spikes in stock prices) or clusters in the data
- Bar plots represent data using rectangular bars where the height or length of each bar corresponds to the value of the data point and are suitable for displaying time series with discrete or aggregated time intervals (quarterly revenue)
- Stem-and-leaf plots combine numerical values and their magnitudes in a single display and are useful for visualizing the distribution of a time series dataset (temperature readings)

Interpretation of time series patterns

- Trend refers to the long-term increase or decrease in the data, which can be linear or non-linear, and is identified by observing the overall direction of the time series (population growth)
- Seasonality involves regular, repeating patterns in the data that can occur at fixed intervals (daily website traffic), and is identified by regularly spaced peaks and troughs in the plot
- Cyclic patterns are recurring patterns that are not fixed to a specific time frame and are often influenced by external factors (business cycles)
- Outliers are data points that significantly deviate from the overall pattern and can be caused by measurement errors, anomalies (sensor malfunctions), or unusual events (natural disasters)

Visualization for time series data

- Smoothing techniques help to reduce noise and highlight underlying patterns: 1. Moving averages calculate the average of a fixed number of consecutive data points (7-day moving average of stock prices) 2. Exponential smoothing assigns exponentially decreasing weights to older data points, emphasizing recent observations while still considering historical data (forecasting product demand)
- Multiple time series plots compare different time series on the same plot using different colors, line styles, or markers to distinguish between series and help identify relationships, correlations, or divergences between series (comparing sales of different products)
- Annotations and labels add informative text, arrows, or highlights to the plot to provide context, explain significant events (product launch), or mark important data points (record-breaking sales), enhancing the interpretability of the plot for the audience

Libraries for time series plotting

- R libraries:
- ggplot2 is a Grammar of Graphics-based plotting library that offers a wide range of customization options and plot types (line plots, scatter plots, bar plots)
- plotly creates interactive and animated plots, enabling zooming, panning, and hovering over data points for additional information (interactive stock price charts)

- Python libraries:
- Matplotlib is a low-level plotting library with high customization capabilities and provides a MATLAB-like plotting interface (customizing plot elements)
- Seaborn is a statistical data visualization library built on top of Matplotlib that offers attractive default styles and color palettes (visualizing distributions)
- Plotly creates interactive and web-based plots, similar to the R version, allowing for zooming, panning, and hovering interactions (interactive dashboards)

1.4 Introduction to R or Python for time series analysis

Time series analysis in R or Python starts with setting up your software and preparing your data. You'll install the necessary tools, load and manipulate your data, and get familiar with key libraries and functions.

Proper data preparation is crucial for accurate analysis. You'll learn to handle missing values, convert data to time series objects, and import/export data in various formats. These skills form the foundation for all your future time series work.

Setting Up and Preparing Data for Time Series Analysis in R or Python

Software setup for time series analysis

- Install R or Python on your computer
- Download and install R from the official website (<https://www.r-project.org/>)
- Download and install Python from the official website (<https://www.python.org/>)
- Set up an Integrated Development Environment (IDE) for coding
- RStudio is a popular IDE for R programming
- Jupyter Notebook, Spyder, or PyCharm are commonly used IDEs for Python programming
- Install required packages for time series analysis
- In R, install packages like 'forecast', 'tseries', 'xts', and 'zoo' using `install.packages()` function
- In Python, install libraries like 'pandas', 'numpy', 'statsmodels', and 'matplotlib' using pip or conda package managers

Data manipulation in R or Python

- Load and inspect time series data
- In R, use functions like `read.csv()`, `read.table()`, or `read.xlsx()` to import data from files (CSV, TXT, Excel)
- In Python, use `read_csv()` or `read_excel()` functions from the pandas library to import data from files (CSV, Excel)
- Handle missing values and outliers
- Identify missing values using functions like `is.na()` in R or `isna()` in Python
- Impute missing values using techniques like mean, median, or last observation carried forward (LOCF)

- Detect and handle outliers using methods like the interquartile range (IQR) or z-score
- Convert data to appropriate time series objects
- In R, convert data to 'ts', 'xts', or 'zoo' objects using functions like `ts()`, `xts()`, or `zoo()`
- In Python, convert data to a pandas 'Series' or 'DataFrame' with a DatetimeIndex using `to_datetime()` and `set_index()` functions

Time series data import/export

- Import data from various file formats
- CSV (Comma-Separated Values) files
- Excel spreadsheets (.xlsx, .xls)
- SQL databases using libraries like 'RSQLite' in R or 'SQLAlchemy' in Python
- Export data to different file formats
- Save processed data as CSV files using `write.csv()` in R or `to_csv()` in Python
- Export data to Excel using the 'writexl' package in R or the 'openpyxl' library in Python
- Handle data with different time frequencies
- Resample data to a desired frequency (daily, monthly, yearly) using functions like `resample()` or `asfreq()` in R
- Resample data to a desired frequency using `resample()` function in Python's pandas library

Libraries for time series analysis

- Familiarize yourself with essential libraries for time series analysis
- In R: 'forecast', 'tseries', 'xts', 'zoo'
- In Python: 'pandas', 'numpy', 'statsmodels', 'matplotlib'
- Understand key functions for time series manipulation and analysis
- In R: `ts()`, `window()`, `lag()`, `diff()`, `acf()`, `pacf()`, `adf.test()`, `Arima()`
- In Python: `shift()`, `diff()`, `autocorr_plot()`, `adfuller()`, `ARIMA()`
- Explore visualization functions for time series data
- In R: `plot()` function, ggplot2 package for creating customizable plots
- In Python: `plot()` function from pandas or matplotlib libraries, `plot_acf()` and `plot_pacf()` functions from statsmodels library

Unit 2 – Time Series Components

2.1 Trend component and its identification

Trend Component and Its Identification

The trend component captures the long-term direction of a time series, whether it's rising, falling, or holding steady over months, years, or decades. Separating the trend from shorter-term noise (like seasonal swings or random spikes) is one of the first steps in any time series analysis, because it tells you the big-picture story your data is telling.

This section covers what trends look like, the main mathematical forms they take, how to detect them, and how to interpret what they mean.

Trend Component Characteristics

A trend reflects the gradual, underlying movement in your data over an extended period. Think of it as the signal beneath the noise. Short-term fluctuations like seasonal patterns or one-off outliers sit on top of the trend, but they don't define it.

A trend can go in one of three directions:

- Upward (increasing): The series generally rises over time
- Downward (decreasing): The series generally falls over time
- Stable (flat): No clear long-term rise or fall

The key word is gradual. Trends are smooth, slow-moving patterns. If something changes sharply from one month to the next, that's probably not the trend; it's a short-term fluctuation or a structural break.

Types of Time Series Trends

Linear Trend

A linear trend means the data changes at a roughly constant rate over time, forming a straight line.

$$Y_t = a + bt$$

Here, a is the y-intercept (the starting level) and b is the slope (how much Y changes per time period). A positive b means the series is rising; a negative b means it's falling.

For example, if a city's population grows by about 5,000 people every year, that's a linear trend. GDP per capita in a steadily developing economy can also follow this pattern.

Exponential Trend

An exponential trend means the data grows (or decays) at a constant percentage rate rather than a constant amount. This produces a curve that accelerates over time.

$$Y_t = ab^t$$

Here, a is the initial value and b is the growth factor. If $b = 1.05$, the series grows 5% each period.

Compound interest is a classic example: your balance doesn't grow by the same dollar amount each year; it grows by the same proportion, so the increases get larger over time. Bacterial growth follows the same logic.

A useful trick: taking the logarithm of exponentially trending data turns the curve into a straight line. This lets you apply linear methods to analyze it.

Polynomial Trend

A polynomial trend captures patterns where the rate of change itself changes over time. The series might accelerate, decelerate, or reverse direction.

$$Y_t = a_0 + a_1t + a_2t^2 + \dots + a_nt^n$$

The degree n controls how many curves the trend line can have. A second-degree polynomial ($n = 2$) produces a parabola, which can model something like a product life cycle: sales rise, peak, then fall.

Choosing between these: Start with the simplest model (linear). Only move to exponential or polynomial if the data clearly curves. Overfitting with a high-degree polynomial is a common mistake; it can follow the noise instead of the trend.

Methods for Trend Detection

Graphical Methods

Time series plot. The simplest first step is to plot your data against time and look at it. You're scanning for a general upward or downward drift across the full time span. This won't give you a precise answer, but it tells you what kind of trend to test for.

Moving average. A moving average smooths out short-term fluctuations so the trend becomes easier to see. Here's how it works:

1. Choose a window size k (e.g., 12 months for monthly data with yearly seasonality).
2. For each time point, calculate the average of the k observations centered on (or preceding) that point.
3. Plot the smoothed values alongside the original data.

The smoothed line reveals the trend by averaging away seasonal and irregular variation. A larger window produces a smoother line but can lag behind real changes in the trend.

Analytical Methods

Least squares regression fits a trend line (linear, exponential, or polynomial) to your data by minimizing the sum of squared differences between observed and predicted values. This gives you an equation you can use for forecasting. For a linear trend, you're estimating a and b in $Y_t = a + bt$.

Mann-Kendall test is a non-parametric test that checks whether a monotonic trend (consistently up or consistently down) exists in the data. It doesn't assume any particular distribution, which makes it useful when your data isn't normally distributed. It tells you whether a trend is statistically significant, but not its exact shape.

Augmented Dickey-Fuller (ADF) test checks whether a time series is stationary (no trend) or non-stationary (contains a trend or unit root). If the test fails to reject the null hypothesis, the series likely has a trend or unit root that needs to be addressed before further modeling.

Interpreting Long-Term Movements

Once you've identified a trend, you need to interpret what it means. This involves three things:

Assess the Direction

1. Increasing trend signals long-term growth or expansion (e.g., global average temperatures rising over decades, stock market indices climbing over the long run).
2. Decreasing trend signals long-term decline (e.g., shrinking natural resource reserves, falling birth rates in many developed countries).
3. Stable trend means no significant long-term shift, though short-term variation may still be present.

Interpret the Slope

The slope tells you the rate of change. A steeper slope means faster change; a shallow slope means slower change. For a linear model, the slope b gives you the exact change per time period. For an exponential model, the growth factor b gives you the percentage change per period.

A positive slope confirms an upward trend; a negative slope confirms a downward trend. Comparing slopes across different time windows can also reveal whether the trend is accelerating or slowing down.

Apply Context and Domain Knowledge

Numbers alone don't tell the full story. Always connect the trend to real-world factors:

- An upward trend in energy consumption might reflect population growth, economic expansion, or both.
- A downward trend in manufacturing output could stem from automation, offshoring, or declining demand.

This context matters for forecasting. A trend driven by a temporary cause (like a policy change) may not continue, while a trend driven by a structural shift (like technological adoption) probably will. Good trend interpretation combines the statistical evidence with an understanding of why the trend exists.

2.2 Seasonal component and patterns

Seasonality in Time Series

Seasonality in Time Series

Seasonality refers to regular, predictable patterns that repeat over a fixed period in your data. Think of it this way: if you plotted monthly ice cream sales over several years, you'd see the same spike every summer and the same dip every winter. That repeating shape is seasonality.

These patterns are driven by factors that consistently influence data at specific times:

- Weather patterns cause predictable swings in energy usage, agriculture, and retail (e.g., winter heating costs rise every December through February)
- Holidays create reliable spikes in spending and travel (e.g., Christmas retail sales, Thanksgiving airline bookings)
- School schedules shape demand cycles (e.g., back-to-school shopping in August, lower activity during summer breaks)

- Business cycles impose structure on financial data (e.g., quarterly earnings reports, year-end budget spending)

Seasonality is one of the most common features in real-world time series. If you ignore it, you risk drawing wrong conclusions. For example, you might mistake a normal summer sales increase for evidence of a growing trend, or you might underestimate demand heading into a holiday season.

Recognition of Seasonal Patterns

The simplest way to spot seasonality is to look at a time series plot. If you see regularly repeating peaks and troughs, that's a strong signal. The distance between consecutive peaks tells you the periodicity, which is the length of one full seasonal cycle.

Common periodicities include:

- Yearly (period = 12 months): Holiday retail sales peak every December; allergy medication sales spike every spring
- Quarterly (period = 3 months): Corporate earnings follow a quarterly reporting cycle; some sports viewership repeats every quarter
- Monthly (period within a month): Utility bills may follow a consistent monthly pattern tied to billing cycles
- Weekly (period = 7 days): Restaurant traffic peaks on weekends; streaming TV viewership dips on weekday mornings

Getting the periodicity right matters. If your data has a yearly cycle but you model it as quarterly, your seasonal adjustments won't line up with the actual pattern, and your forecasts will suffer.

Additive vs. Multiplicative Seasonality

Not all seasonal patterns behave the same way. The two main types differ in how the seasonal effect relates to the overall level of the series.

Additive seasonality means the seasonal swings stay roughly the same size regardless of the series level. If holiday sales always add about 500 units above the trend, whether the baseline is 2,000 or 5,000 units, that's additive. The decomposition model is:

$$Y_t = T_t + S_t + I_t$$

- Y_t : observed value at time t
- T_t : trend component at time t
- S_t : seasonal component at time t
- I_t : irregular (random) component at time t

Multiplicative seasonality means the seasonal swings grow or shrink in proportion to the series level. If holiday sales are always about 20% above the trend, the absolute increase gets larger as the trend rises. The model is:

$$Y_t = T_t \times S_t \times I_t$$

Here, S_t is expressed as a ratio (e.g., 1.20 for 20% above trend, 0.85 for 15% below).

How to tell them apart: Plot your data. If the height of the seasonal peaks stays constant over time, it's additive. If the peaks get taller as the series level increases, it's multiplicative. Choosing the wrong type leads to seasonal adjustments that either over-correct or under-correct your data.

Impact of Seasonality on Forecasting

Seasonality can mask what's really going on in your data. A genuine upward trend might be hard to see when large seasonal swings dominate the plot. Conversely, a flat trend might look like growth if you only compare seasonal peaks.

Seasonal adjustment techniques strip out the seasonal component so you can analyze the underlying trend and irregular movements more clearly:

- Seasonal decomposition methods like STL (Seasonal and Trend decomposition using Loess) and Census X-13 split the series into its trend, seasonal, and irregular parts
- The resulting seasonally adjusted data makes it much easier to spot real changes in direction or level

Forecasting models that explicitly incorporate seasonality produce more accurate predictions:

- Seasonal ARIMA (SARIMA) extends the standard ARIMA framework by adding seasonal autoregressive and moving average terms
- Holt-Winters exponential smoothing captures both trend and seasonality, with separate smoothing parameters for each component

Failing to account for seasonality in your forecasts can lead to systematic errors. You might consistently overpredict during off-peak periods and underpredict during peak ones, which translates directly into poor resource allocation and planning decisions.

2.3 Cyclical and irregular components

Time series analysis involves breaking down data into components. Two key elements are cyclical patterns, which are long-term fluctuations, and irregular components, which are unpredictable variations. Understanding these helps in forecasting and interpreting economic trends.

Cyclical components differ from seasonal patterns in duration and consistency. Irregular components add noise to data. Both affect time series analysis, potentially obscuring trends or seasonality. Techniques like decomposition methods help separate these elements for more accurate analysis and predictions.

Cyclical and Irregular Components in Time Series

Cyclical component characteristics

- Represents recurring fluctuations or oscillations in a time series that occur over periods longer than one year (business cycles, economic conditions)
- Not fixed in frequency or amplitude, meaning the duration and magnitude of the cycles can vary over time
- Wavelength measures the duration of a complete cycle from peak to peak or trough to trough (10-year business cycle)
- Amplitude indicates the magnitude of the fluctuations from the long-term trend ($\pm 2\%$ GDP growth)
- Phase describes the position of the cycle relative to a reference point in time (peak in 2007, trough in 2009)

Seasonal vs cyclical patterns

- Seasonal patterns occur at fixed intervals within a year (quarterly sales, monthly temperatures)
- Have a consistent frequency and amplitude
- Caused by factors such as weather, holidays, or cultural events (summer vacation, Christmas sales)
- Cyclical patterns occur over periods longer than one year
- Have varying frequency and amplitude
- Caused by factors such as business cycles, economic conditions, or technological changes (dot-com boom, housing market crash)

Irregular component in time series

- Represents unpredictable fluctuations in a time series caused by unforeseen events, measurement errors, or other random factors (natural disasters, data entry mistakes)
- Non-systematic, meaning it does not follow a specific pattern or trend
- Short-term, affecting the time series for a brief period
- Assumed to have a zero mean, with fluctuations averaging out to zero over time

Effects of cyclical and irregular components

- Cyclical component effects 1. Introduces long-term oscillations in the time series 2. Can obscure the underlying trend or seasonality 3. May require specialized techniques for extraction and analysis (band-pass filters)
- Irregular component effects
- Adds noise or randomness to the time series
- Can make it difficult to identify and estimate other components (trend, seasonality, cyclical)
- May require smoothing techniques to reduce the impact on the analysis (moving averages)
- Interaction between cyclical and irregular components
- Irregular fluctuations can distort the estimation of the cyclical component
- Cyclical patterns may be harder to detect in the presence of significant irregular variations
- Decomposition methods can help separate the components for more accurate analysis and forecasting (X-11, SEATS)

2.4 Decomposition methods (additive and multiplicative)

Time Series Decomposition

Time series decomposition breaks a complex series into simpler, interpretable parts: trend, seasonality, and irregularity. By isolating each component, you can understand what's driving changes in your data and build better forecasts. The two main approaches, additive and multiplicative, differ in how they assume these components combine.

Purpose of Time Series Decomposition

Decomposition serves three core goals:

- Isolate distinct patterns. A raw time series mixes together long-term growth, seasonal cycles, and random noise. Decomposition separates these so you can study each one independently.
- Understand underlying structure. Once separated, you can see why a series behaves the way it does. For example, are retail sales rising because of a genuine growth trend, or just because the holiday season inflates the numbers every December?
- Improve forecasting. With components isolated, you can apply specialized models to each one (or use the decomposition to inform models like ARIMA or exponential smoothing), rather than trying to model everything at once.

Additive vs. Multiplicative Decomposition Models

The key question is: does the size of seasonal swings stay constant, or does it grow as the series level increases?

Additive decomposition assumes the components simply add together:

$$Y_t = T_t + S_t + I_t$$

where Y_t is the observed value, T_t is the trend, S_t is the seasonal component, and I_t is the irregular (residual) component.

Use additive decomposition when seasonal fluctuations remain roughly the same size regardless of the series level. Think of monthly ice cream sales in a city with stable population: summer bumps up sales by about the same dollar amount each year.

Multiplicative decomposition assumes the components multiply together:

$$Y_t = T_t \times S_t \times I_t$$

Use this when seasonal swings grow (or shrink) proportionally with the trend. A fast-growing tech startup might see Q4 revenue spikes that get larger each year as the company's baseline revenue increases. The seasonal effect here is a ratio, not a fixed amount.

Quick decision rule: Plot your data. If the seasonal peaks and troughs fan out over time (getting wider), use multiplicative. If they stay roughly parallel, use additive.

Application of Additive Decomposition

Here's the step-by-step process:

1.

Estimate the trend (T_t) using a smoothing method. A common choice is a centered moving average. For monthly data, you'd use a 12-month moving average to smooth out seasonal cycles. Alternatively, you can fit a polynomial regression (quadratic, cubic) if the trend has curvature.

2.

Detrend the series by subtracting the trend from the original values: $Y_t - T_t = S_t + I_t$ This removes the long-term direction and leaves only seasonal patterns and noise.

3.

Estimate the seasonal component (S_t) by averaging the detrended values for each season. For monthly data, average all the January detrended values together, all the February values together, and so on. This gives you 12 seasonal indices. These indices are typically adjusted so they sum to zero across a full cycle.

4.

Calculate the irregular component (I_t) by subtracting both the trend and seasonal estimates from the original series: $I_t = Y_t - T_t - S_t$. This residual captures random fluctuations, one-off events, and anything the trend and seasonal components didn't explain.

5.

Verify the decomposition by reconstructing the series: $T_t + S_t + I_t$ should equal Y_t . If it doesn't match closely, something went wrong in your estimation.

Process of Multiplicative Decomposition

The logic mirrors additive decomposition, but you use division instead of subtraction:

1.

Estimate the trend (T_t) using a moving average or regression, same as before.

2.

Detrend the series by dividing the original values by the trend: $\frac{Y_t}{T_t} = S_t \times I_t$. Division preserves the proportional relationship between components, which is the whole point of the multiplicative model.

3.

Estimate the seasonal component (S_t) by averaging the detrended ratios for each season. For quarterly data, average all Q1 ratios, all Q2 ratios, etc. These seasonal indices are typically adjusted so they average to 1.0 across a full cycle (rather than summing to zero, as in additive).

4.

Calculate the irregular component (I_t) by dividing the original series by the product of trend and seasonal estimates: $I_t = \frac{Y_t}{T_t \times S_t}$

5.

Verify by reconstructing: $T_t \times S_t \times I_t$ should reproduce Y_t .

Interpretation of Decomposition Components

Trend (T_t) shows the long-term direction of the series.

- A positive trend means the series is generally increasing (e.g., population growth, rising GDP).
- A negative trend means overall decline (e.g., shrinking market share for a legacy product).
- Shifts or bends in the trend can reveal structural changes, like the onset of a recession or a new competitor entering the market.

Seasonal component (S_t) captures regular, calendar-driven patterns that repeat within a fixed period (usually a year).

- In the additive model, seasonal values are expressed in the same units as the data (e.g., "+3,000 units in December").

- In the multiplicative model, seasonal values are ratios (e.g., "1.15 in December" means 15% above the trend).
- These patterns help with resource planning: a retailer can use seasonal indices to decide how much extra inventory to stock before the holidays.

Irregular component (I_t) is whatever's left after removing trend and seasonality.

- Small, random-looking residuals suggest your decomposition captured the data's structure well.
- Large or clustered residuals may signal outliers, unexpected events (natural disasters, policy changes, supply shocks), or a missing component your model didn't account for.
- Examining the irregular component is also useful for assessing how predictable a series really is. A series with large irregular swings carries more inherent uncertainty for forecasting.

Unit 3 – Stationary vs Non-Stationary Time Series

3.1 Concept of stationarity and its importance

Stationarity in Time Series Analysis

Stationarity is the idea that a time series behaves the same way no matter when you observe it. Its statistical properties don't drift or shift over time. This matters because most classical time series models (ARMA, ARIMA) assume stationarity. If your data violates that assumption, your model's forecasts and parameter estimates can be misleading or flat-out wrong.

This section covers what stationarity actually means, the specific properties that define it, why it matters for modeling, and the distinction between strict and weak stationarity.

Stationarity in Time Series

A time series is stationary when its statistical properties stay constant over time. The mean doesn't trend upward or downward, the variance doesn't expand or shrink, and the correlation structure between observations at different time points doesn't change.

Why does this matter so much?

- Many core models (AR, MA, ARMA) are built on the assumption that the data-generating process is stable. If the process itself is shifting, these models can't reliably learn patterns from historical data and project them forward.
- A non-stationary series can produce spurious relationships, where two variables appear correlated simply because they both trend in the same direction, not because they're actually related.
- Trends, seasonality, or changing variance can all mask the true underlying patterns you're trying to model.

The bottom line: before you fit most time series models, you need to check whether your data is stationary, and if it isn't, you need to transform it until it is.

Properties of a Stationary Series

Three specific properties define (weak) stationarity. All three must hold simultaneously.

1. Constant mean — The expected value doesn't change over time.

$$E(X_t) = \mu \text{ for all } t$$

This means the series fluctuates around the same level throughout. If you see an upward or downward trend, the mean isn't constant.

2. Constant variance — The spread of values stays stable.

$$\text{Var}(X_t) = \sigma^2 \text{ for all } t$$

If the series becomes more volatile in later periods (common with financial data like stock prices), the variance isn't constant.

3. Constant autocovariance — The covariance between any two observations depends only on the lag between them, not on when they occur.

$$\text{Cov}(X_t, X_{t+h}) = \gamma(h) \text{ for all } t \text{ and } h$$

For example, the correlation between observations 1 day apart should be the same whether you measure it in January or July. The function $\gamma(h)$ depends on the lag h , but not on the time index t .

Implications of Stationarity

Stationarity is what allows you to use historical patterns to make predictions about the future. If the statistical behavior of a series keeps changing, past patterns aren't a reliable guide.

Models that rely on stationarity:

1. Autoregressive (AR) models predict the current value using a linear combination of its own past values.
2. Moving Average (MA) models predict the current value using a linear combination of past forecast errors.
3. ARMA models combine both AR and MA components into a single framework.
4. ARIMA models extend ARMA by adding a differencing step (the "I" stands for "Integrated"), which can convert a non-stationary series into a stationary one before fitting the ARMA part.

When your data isn't stationary, you typically need to transform it first:

- Differencing removes trends by computing the change between consecutive observations. For example, instead of modeling raw GDP, you model the change in GDP from one quarter to the next. Seasonal differencing (e.g., subtracting the value from 12 months ago) handles seasonal patterns.
- Variance-stabilizing transformations like logarithmic or square-root transforms can fix non-constant variance. Stock prices, for instance, often have variance that grows with the price level; taking the log compresses that growth.

Strict vs. Weak Stationarity

These are two different levels of the stationarity requirement, and the distinction matters for understanding what models actually assume.

Strict stationarity requires that the entire joint probability distribution of any collection of time points is unchanged when you shift them in time:

$$(X_{t_1}, X_{t_2}, \dots, X_{t_n}) \text{ has the same distribution as } (X_{t_1+h}, X_{t_2+h}, \dots, X_{t_n+h})$$

for all choices of time points and all shifts h . This is a very strong condition. It means every statistical property (not just mean and variance, but skewness, kurtosis, and the full shape of the distribution) must be time-invariant. In practice, strict stationarity is nearly impossible to verify because you'd need complete knowledge of the distribution.

Weak stationarity (also called covariance stationarity) only requires the three properties listed above: constant mean, constant variance, and autocovariance that depends only on lag. This is the version that matters in practice, and it's what people usually mean when they say "stationary" in the context of time series modeling.

For a Gaussian (normally distributed) time series, weak stationarity actually implies strict stationarity. That's because a Gaussian distribution is fully determined by its mean and covariance structure. For

non-Gaussian series, weak stationarity is genuinely a weaker condition.

You can assess weak stationarity through visual inspection (plotting the series and looking for trends or changing spread) and formal statistical tests like unit root tests, which you'll encounter later in this unit.

3.2 Testing for stationarity (visual and statistical methods)

Visual inspection techniques are crucial for assessing stationarity in time series data. By plotting the series and analyzing autocorrelation functions, we can identify trends, seasonality, and changes in variance that indicate non-stationarity.

Statistical tests provide a quantitative approach to confirm or reject visual inspection findings. The Augmented Dickey-Fuller and KPSS tests offer formal hypotheses for determining stationarity, helping us make objective decisions about appropriate modeling techniques.

Visual Inspection Techniques for Stationarity

Visual inspection for stationarity

- Plotting the time series
- Identifies trends, seasonality, and changes in variance over time
- Non-stationary series may exhibit clear trends (upward or downward) or changing variance (increasing or decreasing spread)
- Analyzing the autocorrelation function (ACF)
- Measures the correlation between a time series and its lagged values
- For a stationary series, ACF should decay quickly to zero (within a few lags)
- Slow decay or significant correlations at large lags (beyond the 95% confidence interval) suggest non-stationarity
- Analyzing the partial autocorrelation function (PACF)
- Measures the correlation between a time series and its lagged values, while controlling for the effects of intermediate lags
- For a stationary series, PACF should have significant correlations only at a few lags (typically one or two)
- Persistent significant correlations at multiple lags (beyond the 95% confidence interval) suggest non-stationarity

Interpretation of visual methods

- Plotting the time series
- If the series appears to have a constant mean (no clear trend) and variance (consistent spread) over time, it may be stationary
- If the series exhibits trends (increasing or decreasing), seasonality (regular patterns), or changing variance (increasing or decreasing spread), it is likely non-stationary
- Analyzing the ACF

- If the ACF decays quickly to zero (within a few lags), the series is likely stationary
- If the ACF decays slowly or has significant correlations at large lags (beyond the 95% confidence interval), the series is likely non-stationary
- Analyzing the PACF
- If the PACF has significant correlations only at a few lags (typically one or two), the series is likely stationary
- If the PACF has persistent significant correlations at multiple lags (beyond the 95% confidence interval), the series is likely non-stationary
- Combining visual inspection results
- If all visual inspection methods suggest stationarity (constant mean and variance, quick ACF decay, few significant PACF lags), the series is likely stationary
- If one or more methods suggest non-stationarity (trends, seasonality, changing variance, slow ACF decay, persistent significant PACF lags), further investigation or statistical tests may be necessary

Statistical Tests for Stationarity

Importance of stationarity tests

- Statistical tests provide a formal, quantitative approach to assessing stationarity
- They help to confirm or reject the conclusions drawn from visual inspection methods
- Statistical tests are important for making objective decisions about stationarity and selecting appropriate modeling techniques (ARIMA, SARIMA, etc.)

Hypotheses in stationarity tests

- Augmented Dickey-Fuller (ADF) test
- Null hypothesis: The time series has a unit root and is non-stationary
- Alternative hypothesis: The time series does not have a unit root and is stationary
- If the p-value is less than the chosen significance level (0.05), reject the null hypothesis and conclude that the series is stationary
- Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
- Null hypothesis: The time series is trend-stationary (stationary around a deterministic trend)
- Alternative hypothesis: The time series is not trend-stationary and has a unit root or is non-stationary
- If the p-value is greater than the chosen significance level (0.05), fail to reject the null hypothesis and conclude that the series is trend-stationary

3.3 Differencing and transformation techniques

Differencing and Transformation Techniques

Time series data often needs preprocessing before you can build reliable models. Differencing and transformation techniques are the main tools for converting a non-stationary series into a stationary one. Differencing stabilizes the mean by removing trends and seasonality, while transformations like logarithms stabilize the variance. You'll typically apply transformations first, then differencing.

Concept of Differencing

Many time series models assume stationarity, meaning the statistical properties (mean, variance, autocorrelation) don't change over time. A series with an upward trend or seasonal swings violates this assumption, which makes forecasts unreliable.

Differencing fixes this by computing the change between consecutive observations instead of working with the raw values. This strips out trends and can remove seasonal patterns too.

- The first-order difference subtracts each value from the one before it: $\nabla X_t = X_t - X_{t-1}$
- If one round of differencing isn't enough, you can apply it again (higher-order differencing) until the series looks stationary
- First-order differencing handles linear trends well. More complex trends (quadratic, exponential growth) may need additional rounds.

First-Order Differencing Application

First-order differencing is by far the most common form you'll encounter. The calculation is simple:

$$\nabla X_t = X_t - X_{t-1}$$

Each value in the differenced series represents the change from one time step to the next. A positive value means the series went up; a negative value means it went down. The magnitude tells you how fast it changed.

For example, if monthly sales go from 200 to 230, the differenced value is 30, capturing that month's growth.

This works well when the original series has a roughly linear trend. After differencing, that upward or downward drift disappears, leaving a series that fluctuates around a constant mean. However, first-order differencing only addresses the mean. If the variance is also changing over time (heteroscedasticity), or if there's seasonality, you'll need additional techniques.

Higher-Order Differencing

Sometimes first-order differencing isn't enough. Two common situations call for going further:

Second-order differencing applies the differencing operation to an already-differenced series:

$$\nabla^2 X_t = \nabla X_t - \nabla X_{t-1}$$

This is useful when the original trend is nonlinear, like a quadratic curve. Think of it this way: first-order differencing captures the rate of change, and second-order differencing captures the change in the rate of change (similar to acceleration vs. velocity in physics).

Seasonal differencing removes repeating seasonal patterns by subtracting the value from the same season in the previous cycle:

$$\nabla_s X_t = X_t - X_{t-s}$$

Here, s is the seasonal period. For monthly data with a yearly cycle, $s = 12$. For quarterly data, $s = 4$. So with monthly data, January's value gets subtracted from the following January's value, February from February, and so on.

A word of caution: overdifferencing is a real pitfall. Each round of differencing removes information from the series and can introduce artificial patterns. Only difference as many times as needed. Use visual inspection (does the differenced series look stationary?) and formal tests like the Augmented Dickey-Fuller test to decide.

Logarithmic and Power Transformations

Differencing stabilizes the mean, but many series also have non-constant variance. A classic example: retail sales where the seasonal swings get bigger as overall sales grow. The December spike might be ± 50 units in the early years but ± 500 units later. This is heteroscedasticity, and it violates model assumptions.

Logarithmic transformation handles this by compressing large values more than small ones:

$$y_t = \log(x_t)$$

Use this when variance increases proportionally with the level of the series (multiplicative patterns). After taking the log, differences in the transformed series approximate percentage changes, which is often a more natural way to think about growth.

Power transformations generalize the log transform. The Box-Cox transformation uses a parameter λ to find the best transformation:

$$y_t = \frac{x_t^\lambda - 1}{\lambda} \quad \text{for } \lambda \neq 0, \quad y_t = \log(x_t) \quad \text{for } \lambda = 0$$

The value of λ is estimated from the data to minimize variance in the transformed series. Some common special cases:

- $\lambda = 1$: no transformation (original data)
- $\lambda = 0.5$: square root
- $\lambda = 0$: natural log
- $\lambda = -1$: reciprocal

The standard workflow is: transform first, then difference. You apply the log or Box-Cox transformation to stabilize variance, and then difference the transformed series to stabilize the mean. This two-step process gives you a series with both constant mean and constant variance, ready for modeling with methods like ARIMA.

3.4 Unit root tests (ADF, KPSS)

Unit Root Tests

Unit root tests tell you whether a time series is stationary or non-stationary. This matters because non-stationary data can produce spurious regressions, where two unrelated series appear to have a significant relationship simply because they both trend over time. Before you model a time series, you need to know what you're working with.

The two main tests covered here are the Augmented Dickey-Fuller (ADF) test and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test. They approach the question from opposite directions: ADF assumes non-stationarity until proven otherwise, while KPSS assumes stationarity until proven otherwise. Using both together gives you a much more reliable answer.

Unit roots and time series stationarity

A unit root is a feature of a time series where shocks don't fade away over time. Instead, they permanently shift the level of the series. A classic example is a random walk: each value equals the previous value plus some random noise, so any shock gets "baked in" forever.

Why does this matter for stationarity?

- A stationary series has a constant mean, constant variance, and autocovariance that depends only on the lag (not on time). It tends to revert to its mean after a shock.
- A series with a unit root is non-stationary. Its mean and variance change over time, and it can drift upward or downward without bound.
- If you run a regression on non-stationary data without addressing the unit root, you risk finding relationships that don't actually exist. Standard t-tests and F-tests become unreliable.

Application of ADF test

The Augmented Dickey-Fuller (ADF) test checks whether a unit root is present. Its null hypothesis is that the series has a unit root (is non-stationary). You're trying to gather enough evidence to reject that null.

The "augmented" part means it extends the original Dickey-Fuller test by including lagged difference terms. These extra terms absorb any serial correlation in the residuals so the test statistic remains valid.

The ADF regression equation:

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^p \delta_i \Delta y_{t-i} + \varepsilon_t$$

Here's what each piece does:

- Δy_t : The first difference of the series (today's value minus yesterday's)
- α : A constant (intercept), allowing for a non-zero mean
- βt : A deterministic time trend (you can include or exclude this depending on whether you think the series has a trend)
- γy_{t-1} : The lagged level of the series. This is the key term. If $\gamma = 0$, the series has a unit root.
-

$$\sum_{i=1}^p \delta_i \Delta y_{t-i}$$

: Lagged differences that capture short-run dynamics and clean up autocorrelation. You choose the lag order p using information criteria (like AIC or BIC).

Steps to run the ADF test:

1. Decide whether to include a constant only, or a constant plus trend, based on what the data looks like.

2. Choose the lag order p (often selected automatically by AIC/BIC).
3. Estimate the regression above using OLS.
4. Compute the t-statistic for γ .
5. Compare that t-statistic to the ADF critical values (these are not standard t-distribution critical values; they come from special Dickey-Fuller tables).

Interpretation of ADF test results

The ADF test statistic is a t-ratio for γ , but you compare it against Dickey-Fuller critical values, not the usual t-distribution. These critical values are more negative than standard ones.

- If the test statistic is more negative than the critical value at your chosen significance level (1%, 5%, or 10%), you reject the null. The series is stationary.
- If the test statistic is less negative than the critical value, you fail to reject the null. The evidence suggests a unit root is present.

For example, suppose your ADF statistic is -3.82 and the 5% critical value is -2.89 . Since $-3.82 < -2.89$, you reject the null and conclude the series is stationary.

When you fail to reject, the typical next step is to difference the series (take Δy_t) and test again. If the differenced series is stationary, you say the original series is "integrated of order 1," or $I(1)$.

KPSS test for stationarity

The KPSS test flips the logic. Its null hypothesis is that the series is trend-stationary (or level-stationary, depending on the version you use). The alternative is that a unit root is present.

This reversal is useful because the ADF test's failure to reject doesn't confirm non-stationarity; it just means you didn't find enough evidence against it. The KPSS test lets you actively test for stationarity.

Steps to run the KPSS test:

1. Regress the series on a constant (level-stationarity version) or a constant plus time trend (trend-stationarity version).
 2. Collect the residuals from that regression.
 3. Compute the KPSS test statistic from the cumulative sum of those residuals. The statistic is based on a ratio involving the partial sum of residuals and an estimate of the long-run variance.
 4. Compare the test statistic to KPSS critical values.
- A large test statistic means you reject the null of stationarity, suggesting the series has a unit root.
 - A small test statistic means you fail to reject, supporting stationarity.

One practical detail: the KPSS test requires choosing a bandwidth (lag truncation parameter) to estimate the long-run variance. Results can be sensitive to this choice, so it's worth checking whether your conclusion changes with different bandwidth settings.

ADF vs KPSS test comparison

The key difference between these tests is their null hypotheses:

	ADF	KPSS
Null hypothesis	Unit root (non-stationary)	Trend-stationary
Alternative hypothesis	Stationary	Unit root (non-stationary)
Reject null means	Series is stationary	Series is non-stationary

Strengths and limitations:

- The ADF test is widely used and straightforward to interpret, but it has low power against near-unit-root processes. If the true autoregressive coefficient is close to 1 (say, 0.97), ADF often fails to reject the null even though the series is technically stationary. It also struggles with small samples.
- The KPSS test is more sensitive at detecting departures from stationarity, but it's sensitive to the bandwidth parameter. Different bandwidth choices can lead to different conclusions.

Using both tests together gives you four possible outcomes:

ADF result	KPSS result	Conclusion
Rejects null (stationary)	Fails to reject null (stationary)	Strong evidence of stationarity
Fails to reject (unit root)	Rejects null (non-stationary)	Strong evidence of non-stationarity
Rejects null (stationary)	Rejects null (non-stationary)	Conflicting results; may need more investigation
Fails to reject (unit root)	Fails to reject (stationary)	Conflicting results; may need more investigation

When both tests agree, you can be more confident in your conclusion. When they conflict, the series may be near the boundary between stationary and non-stationary, and you might need a longer sample, a different test (like the Phillips-Perron test), or careful judgment about how to proceed.

Unit 4 – Autocorrelation and Partial Autocorrelation

4.1 Autocorrelation function (ACF) and its interpretation

Autocorrelation Function (ACF)

The Autocorrelation Function (ACF) measures how correlated a time series is with its own past values at different time lags. It's one of the first diagnostic tools you'll reach for when analyzing time series data, because the shape of the ACF plot tells you a lot about what kind of model might fit your data well.

Definition and Purpose of ACF

Autocorrelation is the correlation between a time series and a lagged copy of itself. The ACF computes this correlation at every lag (1, 2, 3, ...) and gives you a complete picture of how the series relates to its own history.

The ACF at lag k is defined as:

$$r_k = \frac{\sum_{t=k+1}^T (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^T (y_t - \bar{y})^2}$$

where y_t is the observation at time t , $\bar{y}$ is the series mean, and T is the total number of observations. The result is always between -1 and $+1$, just like a regular correlation coefficient.

The ACF serves several purposes:

- Pattern detection: A slowly decaying ACF suggests a trend. Spikes at regular intervals point to seasonality.
- Model identification: The shape of the ACF helps you decide what type of time series model (AR, MA, ARMA) is appropriate.
- Residual diagnostics: After fitting a model, you check the ACF of the residuals. If significant autocorrelation remains, the model hasn't captured all the structure in the data.

Interpretation of ACF Plots

An ACF plot (also called a correlogram) shows the correlation coefficient on the y-axis and the lag number on the x-axis. Two horizontal dashed lines mark the 95% confidence bounds, typically at $\pm \frac{1.96}{\sqrt{T}}$. Any spike that extends beyond these bounds is statistically significant.

Here's how to read common ACF patterns:

Positive autocorrelation shows up as ACF values that stay positive and decay slowly toward zero. This means high values tend to follow high values, and low values tend to follow low values. You'll see this in data with trends or strong momentum.

Negative autocorrelation produces ACF values that alternate in sign, flipping between positive and negative at successive lags. This indicates an oscillating pattern where a high value tends to be followed by a low value and vice versa.

No autocorrelation (white noise) looks like ACF values hovering near zero at all lags, with at most one or two spikes barely crossing the confidence bounds by chance. This is what you want to see in model residuals.

Trend causes the ACF to decay very slowly, staying significant for many lags. The series has long memory, meaning distant past values still correlate with the present.

Seasonality produces noticeable spikes at regular intervals. For monthly data with an annual cycle, you'd see spikes at lags 12, 24, 36, etc. The ACF at lag 12 captures the correlation between, say, this January and last January.

Significance of Lag in ACF

The lag is simply the number of time steps separating two observations being compared.

- Lag 1: correlation between each observation and the one immediately before it
- Lag 2: correlation between each observation and the one two steps back
- Lag k : correlation between observations separated by k time steps

Why lag matters for understanding your data:

- Slowly decaying ACF across many lags means the series has long-term dependence. Current values are influenced by the distant past, and the data likely needs differencing before modeling.
- Quickly decaying ACF (drops to insignificance within a few lags) means short-term dependence. Only recent past values matter for predicting the present.
- The lag at which spikes appear tells you the period of any seasonal pattern. Significant ACF at lag 7 in daily data, for instance, suggests a weekly cycle.

Application of ACF for Model Selection

The ACF is especially useful for identifying Moving Average (MA) models. An MA model of order q uses past forecast errors to predict the current value:

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

A pure MA(q) process has a distinctive ACF signature: significant spikes at lags 1 through q , then a sharp cutoff to zero. So if the ACF drops to insignificance after lag 2, that suggests an MA(2) model.

For Autoregressive (AR) models, the ACF alone isn't enough. An AR(p) model uses past values of the series itself:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

The ACF of an AR process decays gradually (exponentially or in a damped oscillation) rather than cutting off sharply. To pin down the order p , you need the Partial Autocorrelation Function (PACF), which removes the influence of intermediate lags. The PACF of an AR(p) process cuts off sharply after lag p .

Here's a quick reference for using ACF and PACF together:

Pattern	ACF	PACF	Suggested Model
Gradual decay	Tails off slowly	Cuts off after lag p	AR(p)
Sharp cutoff	Cuts off after lag q	Tails off slowly	MA(q)
Both tail off	Tails off	Tails off	ARMA(p, q)

Steps for using the ACF in model selection:

1. Plot the ACF and check for trend or seasonality. If the ACF decays very slowly, difference the series first and recompute.
2. Look at the shape of the ACF. A sharp cutoff suggests an MA model; gradual decay suggests an AR or ARMA model.
3. Plot the PACF alongside the ACF. Use both together to narrow down the model type and order.
4. Fit the candidate model, then check the ACF of the residuals. If no significant spikes remain, the model is adequate.

4.2 Partial autocorrelation function (PACF) and its interpretation

Partial Autocorrelation Function (PACF)

The Partial Autocorrelation Function (PACF) measures the correlation between a time series observation and a lagged version of itself, after removing the effects of all intermediate lags. While the regular ACF shows the total correlation at each lag (which includes indirect effects passed through shorter lags), the PACF isolates the direct relationship at each lag. This distinction makes the PACF essential for identifying how many autoregressive (AR) terms belong in an ARIMA model.

PACF Definition

The core idea: when you calculate the correlation between Y_t and Y_{t-k} , some of that correlation might just be because Y_t is correlated with Y_{t-1} , which is correlated with Y_{t-2} , and so on down to Y_{t-k} . The PACF strips away those indirect, "passed-along" correlations.

- The PACF at lag k is the correlation between Y_t and Y_{t-k} after removing the linear effects of lags $1, 2, \dots, k-1$
- Think of it like a regression: if you regressed Y_t on $Y_{t-1}, Y_{t-2}, \dots, Y_{t-k}$, the PACF at lag k would be the coefficient on Y_{t-k}
- This "controlling for intermediate lags" is what makes the PACF useful for pinpointing the order of an AR model

PACF Plot Interpretation

A PACF plot shows the partial autocorrelation value on the y-axis for each lag on the x-axis, along with confidence bands (typically at the 95% level). Here's how to read it:

- Sharp cutoff pattern: For a pure $AR(p)$ process, the PACF will show significant spikes at lags $1, 2, \dots, p$ and then drop to approximately zero for all lags beyond p . The lag where it cuts off tells you the AR order. For example, if the PACF is significant at lags 1 and 2 but insignificant from lag 3 onward, that points to an $AR(2)$ model.
- Gradual decay pattern: If the PACF doesn't cut off sharply but instead tapers off gradually (exponentially or in a damped sinusoidal pattern), this suggests a moving average (MA) component rather than a pure AR process. You'd then turn to the ACF plot to identify the MA order.
- Significance bands: Spikes that fall within the confidence bands (the shaded region or dashed lines) are treated as not significantly different from zero. Only spikes that extend beyond these bands count as meaningful.

ACF vs. PACF Comparison

Using ACF and PACF plots together is how you identify the structure of an ARIMA model. Each plot reveals different components:

Pattern	ACF Behavior	PACF Behavior
Pure AR(p)	Gradual decay (tails off)	Sharp cutoff after lag p
Pure MA(q)	Sharp cutoff after lag q	Gradual decay (tails off)
ARMA(p, q)	Gradual decay	Gradual decay

The ACF is better at identifying the MA order (q) because it cuts off sharply for pure MA processes. The PACF is better at identifying the AR order (p) because it cuts off sharply for pure AR processes. When both plots show gradual decay, you're likely dealing with a mixed ARMA model, and determining the exact orders requires more careful analysis (such as comparing information criteria like AIC or BIC across candidate models).

PACF Behavior for MA Models

The section heading says "moving average models," so here's the key point: for a pure MA process, the PACF does not cut off sharply. Instead, it decays gradually, often showing an exponential or oscillating tail-off pattern. This is the mirror image of how the ACF behaves for pure AR models.

- If you see gradual decay in the PACF combined with a sharp cutoff in the ACF at lag q , that suggests an MA(q) model
- If the PACF decays gradually and the ACF also decays gradually, consider a mixed ARMA model
- In practice, you'd use both plots together: the ACF cutoff identifies the MA order, while the PACF cutoff identifies the AR order

A practical example: suppose the ACF cuts off after lag 1 and the PACF shows damped exponential decay. This pattern suggests an MA(1) model. If you instead saw the PACF cut off after lag 2 while the ACF decayed gradually, that would suggest an AR(2) model.

4.3 Ljung-Box test and white noise processes

White Noise Processes and Model Validation

White noise is the baseline of time series analysis. It represents purely random data with no patterns, trends, or correlations. When you fit a time series model, your goal is for the residuals (the leftover errors) to look like white noise. If they do, your model has captured everything meaningful in the data. If they don't, something systematic is still hiding in the residuals.

The Ljung-Box test is the standard tool for checking whether residuals behave like white noise. It tests for the presence of autocorrelation across multiple lags at once, giving you a single, clear verdict on model adequacy.

Properties of White Noise Processes

A white noise process is a sequence of uncorrelated random variables with constant mean and variance. Formally, a process $\{\varepsilon_t\}$ is white noise if it satisfies three conditions:

- Zero mean: $E(\varepsilon_t) = 0$ for all time periods t

- Constant variance: $\text{Var}(\varepsilon_t) = \sigma^2$ for all t (this doesn't change over time)
- Zero covariance: $\text{Cov}(\varepsilon_t, \varepsilon_s) = 0$ for all $t \neq s$ (no correlation between observations at different time points)

Because there's no correlation between any two time points, both the autocorrelation function (ACF) and the partial autocorrelation function (PACF) equal zero at every lag except lag 0. On a correlogram, white noise looks like random spikes that stay within the confidence bands. There are no predictable patterns or trends.

Application of the Ljung-Box Test

The Ljung-Box test checks whether a group of autocorrelations in the residuals are collectively different from zero. Rather than testing each lag individually, it pools information across multiple lags into one test statistic.

Hypotheses:

- Null hypothesis (H_0): The residuals are independently distributed (no autocorrelation at any of the tested lags).
- Alternative hypothesis (H_a): At least some autocorrelation exists in the residuals, meaning the model hasn't fully captured the data's dependence structure.

Test statistic:

$$Q = n(n+2) \sum_{k=1}^h \frac{\hat{\rho}_k^2}{n-k}$$

where:

- n = sample size (number of residuals)
- h = number of lags being tested
- $\hat{\rho}_k$ = sample autocorrelation at lag k

Under the null hypothesis, Q follows a chi-squared distribution with h degrees of freedom. Larger values of Q mean more total autocorrelation was detected across the tested lags.

Choosing h : A common rule of thumb is to set h around 10 for non-seasonal data, or around 20 for seasonal data (e.g., $2m$ where m is the seasonal period). Too few lags and you might miss autocorrelation at longer horizons; too many and the test can lose power.

Interpretation of Ljung-Box Results

Once you've computed Q , compare it to the chi-squared critical value at your chosen significance level (typically 0.05):

- If Q exceeds the critical value (or equivalently, the p-value is below 0.05): Reject H_0 . The residuals show significant autocorrelation, which means your model is missing some structure in the data. You should consider modifying the model, perhaps by adding more AR or MA terms, including seasonal components, or investigating other explanatory variables.
- If Q is less than the critical value (p-value above 0.05): Fail to reject H_0 . The residuals are consistent with white noise, supporting the conclusion that your model has adequately captured the autocorrelation structure.

One thing to watch: failing to reject the null doesn't prove the residuals are white noise. It just means the test didn't find enough evidence to say otherwise. With small sample sizes, the test may lack the power to detect weak autocorrelation.

Model Validation and White Noise Residuals

Why White Noise Validation Matters

White noise residuals are the clearest sign of a well-fitted time series model. They indicate that the model has extracted all systematic information from the data, leaving behind only random noise. This also supports the model's underlying assumptions, like independence and constant variance, which matter for producing reliable forecasts and valid confidence intervals.

If the residuals are not white noise, that's a signal to dig deeper. Common issues include:

- **Unmodeled autocorrelation:** The residuals still show trends or cyclical patterns, meaning the model needs additional lags or different AR/MA orders.
- **Missing explanatory variables:** External factors that influence the series haven't been included.
- **Structural breaks or outliers:** Sudden shifts in the data (like a policy change or economic shock) that the model doesn't account for, which can bias your parameter estimates.

In practice, you'll typically combine the Ljung-Box test with visual inspection of the ACF and PACF plots of the residuals. The test gives you a formal statistical answer, while the plots help you see where the remaining autocorrelation is, guiding you toward specific model improvements.

Unit 5 – Moving Average & Exponential Smoothing

5.1 Simple and weighted moving averages

Moving Averages

Simple moving averages calculation

A simple moving average (SMA) smooths out short-term fluctuations by taking the average of a fixed number of past observations. That fixed number is called the window size or order, often written as k .

The formula for the SMA of order k at time t :

$$SMA_t = \frac{1}{k} \sum_{i=0}^{k-1} y_{t-i}$$

where y_{t-i} is the observed value at time $t-i$. Every observation in the window gets equal weight of $\frac{1}{k}$.

Example: 3-period SMA

Given observations: $y_1 = 10, y_2 = 12, y_3 = 8, y_4 = 14$

1. At $t = 3$: $SMA_3 = \frac{1}{3}(10 + 12 + 8) = 10$
2. At $t = 4$: $SMA_4 = \frac{1}{3}(12 + 8 + 14) = 11.33$

Notice that each new SMA value drops the oldest observation and adds the newest one. This "sliding window" is what makes it a moving average.

Weighted moving averages computation

A weighted moving average (WMA) lets you assign different weights to past observations, typically giving more importance to recent values. The key constraint is that all weights must sum to 1.

The formula for the WMA of order k at time t :

$$WMA_t = \sum_{i=0}^{k-1} w_i y_{t-i}$$

where w_i is the weight for the observation at time $t-i$, and $\sum w_i = 1$.

Example: 3-period WMA with weights $w_0 = 0.5, w_1 = 0.3, w_2 = 0.2$

Given the same observations: $y_1 = 10, y_2 = 12, y_3 = 8, y_4 = 14$

1. At $t = 3$: $WMA_3 = 0.5(8) + 0.3(12) + 0.2(10) = 9.6$
2. At $t = 4$: $WMA_4 = 0.5(14) + 0.3(8) + 0.2(12) = 11.8$

Compare $WMA_4 = 11.8$ to $SMA_4 = 11.33$. The WMA reacts more strongly to the recent jump to 14 because it places half the total weight on the most recent observation. This is the core trade-off: a WMA tracks recent changes more closely, but it's also more sensitive to noise.

Advantages vs limitations of moving averages

- Advantages
 - Simple to understand and implement
 - Smooths out short-term fluctuations, making underlying trends easier to see
 - Helps identify trend direction and potential reversals
- Limitations
 - Always a lagging indicator because it relies entirely on past data
 - Can miss sudden shifts like outliers or structural breaks
 - Sensitive to the choice of order k : too small and you get noisy output, too large and you over-smooth real changes
 - Does not account for seasonality, cyclical patterns, or other complex structure on its own

Order selection for moving averages

Choosing the right k is about balancing smoothness against responsiveness. There's no single correct answer; it depends on your data and your goal.

Factors to consider:

1. Seasonality: If your data has a seasonal cycle, set k equal to the seasonal period (e.g., $k = 12$ for monthly data with a yearly cycle). This averages out the seasonal effect.
2. Noise level: Noisier data benefits from a larger k to smooth out random variation.
3. Trend stability: If the trend is steady, a larger k works well. If the trend shifts frequently, a smaller k keeps you closer to the current behavior.
4. Responsiveness: Smaller orders react faster to recent changes but smooth less effectively.

In practice, you should try several values of k and compare results. Use out-of-sample testing (hold back some data, forecast it, and measure error) to pick the order that generalizes best rather than just fitting the historical data well.

5.2 Simple exponential smoothing

Simple Exponential Smoothing

Simple exponential smoothing (SES) is a forecasting technique that weights recent observations more heavily than older ones. It works well for short-term forecasts when your data has no clear trend or seasonal pattern, essentially smoothing out noise to reveal the underlying level of a series.

The method relies on a single parameter, alpha (α), to control how much influence new data has versus old data. By adjusting α , you can make forecasts more reactive to recent changes or more stable over time.

Concept of Exponential Smoothing

The core idea is that not all past observations deserve equal attention. SES assigns exponentially decreasing weights to older data points, so the most recent observation has the strongest influence on your forecast, the one before it has less, the one before that even less, and so on.

For example, if $\alpha = 0.3$, the most recent observation gets a weight of 0.3, the one before it gets roughly $0.3 \times 0.7 = 0.21$, the next gets about $0.3 \times 0.7^2 = 0.147$, and so on. The weights shrink quickly, which is why it's called exponential smoothing.

This approach is best suited for stationary time series, meaning data that fluctuates around a roughly constant level without trending upward/downward or showing seasonal cycles. Common use cases include daily sales figures or weekly website traffic where you just need a near-term forecast.

Application of Simple Exponential Smoothing

The SES forecast formula is:

$$\hat{y}_{t+1} = \alpha y_t + (1 - \alpha) \hat{y}_t$$

where:

- $\hat{y}_{t+1}$ = forecast for the next period
- y_t = actual observed value at time t
- $\hat{y}_t$ = forecast for time t (your previous forecast)
- α = smoothing parameter, a value between 0 and 1

In plain terms: your next forecast is a blend of what actually just happened (y_t) and what you previously predicted ($\hat{y}_t$). The parameter α controls the mix.

How to apply SES step by step:

1. Initialize. Set the first forecast equal to the first observed value: $\hat{y}_1 = y_1$.
2. Choose a value for α (or optimize it later using error measures).
3. Compute each forecast by plugging the current actual value and the previous forecast into the formula.
4. Repeat for each time period in your series.

Suppose your first three observations are 100, 110, and 105, and $\alpha = 0.4$:

- $\hat{y}_1 = 100$ (initialization)
- $\hat{y}_2 = 0.4(100) + 0.6(100) = 100$
- $\hat{y}_3 = 0.4(110) + 0.6(100) = 104$
- $\hat{y}_4 = 0.4(105) + 0.6(104) = 104.4$

Notice how the forecast adjusts gradually toward each new observation rather than jumping to it.

Impact of the Smoothing Parameter

The value of α is the single most important choice you make with SES.

- High α (closer to 1): The forecast leans heavily on the most recent observation. This makes it responsive to sudden shifts in level, but also more sensitive to random noise.
- Low α (closer to 0): The forecast relies more on the accumulated history. This produces smoother, more stable forecasts, but it reacts slowly when the true level of the series changes.

A practical way to think about it: volatile data where the level genuinely shifts often may benefit from a higher α , while stable data with lots of random noise usually calls for a lower α .

Finding the optimal α : Rather than guessing, you can try many values of α (from 0 to 1) and pick the one that minimizes a forecast error measure like the sum of squared errors (SSE) over your historical data. Most time series software does this automatically.

Accuracy of Exponential Smoothing Forecasts

To judge how well SES performs, you compare its forecasts against actual values using error measures:

1. Mean Absolute Error (MAE): The average of the absolute differences between actual values and forecasts. Easy to interpret since it's in the same units as your data.
2. Mean Squared Error (MSE): The average of the squared differences. Squaring penalizes large errors more heavily than small ones, so MSE is useful when big misses are especially costly.
3. Mean Absolute Percentage Error (MAPE): The average of the absolute percentage errors. Useful for comparing accuracy across series with different scales (e.g., a series in the hundreds vs. one in the thousands).

For all three, lower values mean better accuracy. You can also use these measures to compare SES against other methods like a simple moving average. If SES produces a lower MAE than a 5-period moving average on the same data, that's evidence SES is the better fit for that particular series.

5.3 Holt's linear trend method

Holt's Linear Trend Method

Simple exponential smoothing works well for data that fluctuates around a roughly constant level, but it falls behind when your data has a clear upward or downward trend. Holt's linear trend method solves this by adding a second equation that explicitly tracks the trend. This makes it a go-to technique for forecasting things like steadily growing sales or declining inventory levels.

The method relies on two smoothing equations (level and trend) plus a forecast equation. Understanding how these three pieces fit together is the core of this topic.

How Holt's Method Incorporates Trend

Simple exponential smoothing uses one equation to estimate the level of a series. Holt's method extends this with a second equation that estimates the trend (the amount the series increases or decreases per period).

Level equation:

$$\ell_t = \alpha y_t + (1 - \alpha)(\ell_{t-1} + b_{t-1})$$

This updates the estimated level at time t . Notice the term $\ell_{t-1} + b_{t-1}$ in parentheses: that's last period's level plus last period's trend, which together form the one-step-ahead forecast from the previous period. So the level equation is a weighted average of the new observation y_t and the previous forecast.

- α is the level smoothing parameter, where $0 \leq \alpha \leq 1$

Trend equation:

$$b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1}$$

This updates the estimated trend at time t . The term $l_t - l_{t-1}$ is the observed change in level, and b_{t-1} is the previous trend estimate. So the trend equation is a weighted average of the most recent level change and the previous trend estimate.

- β is the trend smoothing parameter, where $0 \leq \beta \leq 1$

Initializing the recursion:

Both equations are recursive, so you need starting values l_0 and b_0 . Common approaches:

- Fit a simple linear regression to the first few observations (say 3–5 points). The intercept gives l_0 and the slope gives b_0 .
- Set l_0 to the first observation y_1 and b_0 to $y_2 - y_1$ (or the average of the first few differences).

Parameter Estimation

You need to choose values for α and β . The standard approach is to pick the combination that minimizes a forecast error measure, most commonly the sum of squared errors (SSE):

$$SSE = \sum_{t=1}^n (y_t - \hat{y}_t)^2$$

where $\hat{y}_t$ is the one-step-ahead forecast for time t .

Two main ways to find optimal parameters:

- Grid search: Try every combination of α and β over a grid (e.g., 0.01, 0.02, ..., 0.99) and keep the pair with the lowest SSE. Simple but can be slow with fine step sizes.
- Numerical optimization: Use an algorithm (like Nelder-Mead or L-BFGS-B) to search for the minimum more efficiently. This is what most software packages do by default.

You can also minimize MAE or MAPE instead of SSE, depending on what kind of errors matter most for your application.

Forecasting with Holt's Method

Once you've estimated l_t and b_t up through the most recent observation, generating forecasts is straightforward:

$$\hat{y}_{t+h|t} = l_t + hb_t$$

Here h is the forecast horizon (how many periods ahead you want to predict). The forecast is just the current level plus h times the current trend. This produces a straight line extending from the last estimated level with slope b_t .

For example, if $l_{20} = 150$ and $b_{20} = 3$, then: - Forecast for period 21: $150 + 1(3) = 153$ - Forecast for period 23: $150 + 3(3) = 159$

Assessing forecast accuracy uses the standard error measures:

- MAE:

$$\frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t|$$

(average size of errors in original units)

- MSE:

$$\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2$$

(penalizes large errors more heavily)

- MAPE:

$$\frac{1}{n} \sum_{t=1}^n \left| \frac{y_t - \hat{y}_t}{y_t} \right| \times 100\%$$

(percentage-based, useful for comparing across different scales)

Residual analysis is also important. Plot the residuals ($y_t - \hat{y}_t$) over time and check the autocorrelation function (ACF) of the residuals. If you see patterns or significant autocorrelations, the model isn't capturing all the structure in the data, and a different method might be needed.

Holt's Method vs. Simple Exponential Smoothing

The key distinction is simple: Holt's method is designed for data with a trend, while simple exponential smoothing assumes no trend.

- Simple exponential smoothing models only the level. If your data has a trend, the forecasts will systematically lag behind because they can't "keep up" with the upward or downward movement.
- Holt's method captures both level and trend, so it can project the trajectory forward rather than just flattening out.

When deciding between them:

- Compare accuracy measures (MAE, MSE, MAPE) on the same dataset. Lower values indicate better fit.
- Use time series cross-validation (rolling origin or expanding window) to test how each method performs on multiple hold-out sets. This gives a more reliable comparison than a single train/test split.
- Consider the trade-off between accuracy and simplicity. Holt's method has two parameters instead of one, which adds a small amount of complexity. If the data has no real trend, the extra parameter just adds noise to the estimation, and simple exponential smoothing will often perform just as well or better.

A practical rule: plot your data first. If you see a clear upward or downward movement over time, Holt's method is the better choice. If the data hovers around a stable level, stick with simple exponential smoothing.

5.4 Holt-Winters' seasonal method

Holt-Winters' Seasonal Method

The Holt-Winters' seasonal method forecasts time series data that contain both a trend and a repeating seasonal pattern. It builds on exponential smoothing by tracking three components at once: the level, the trend, and the seasonal effect. If you've worked with simple or double exponential smoothing, Holt-Winters' is the natural next step for data like monthly retail sales or quarterly energy demand, where ignoring seasonality would produce poor forecasts.

Holt-Winters' method for seasonality

Holt-Winters' works by maintaining three smoothing equations that update at each time step:

- Level equation — estimates the baseline value of the series after removing trend and seasonal effects
- Trend equation — estimates the current slope (how much the series rises or falls per period)
- Seasonal equation — estimates the seasonal factor for each season (e.g., one factor per month if you have monthly data)

Each equation has its own smoothing parameter:

- α (level): controls how much weight goes to the most recent observation vs. the previous level estimate. Range: $0 < \alpha < 1$.
- β (trend): controls how quickly the trend estimate reacts to changes. Range: $0 < \beta < 1$.
- γ (seasonal): controls how quickly seasonal factors update. Range: $0 < \gamma < 1$.

Together, these three equations let the model adapt to shifts in the baseline, changes in the growth rate, and evolving seasonal patterns, all within a single framework.

Additive vs. multiplicative seasonality

Choosing between additive and multiplicative seasonality is one of the most important modeling decisions you'll make with Holt-Winters'.

Additive seasonality assumes the seasonal swings stay roughly the same size no matter how high or low the series is. The seasonal component is added to the level and trend. Use this when a time series plot shows seasonal peaks and troughs of consistent height over time. For example, if a city's monthly bus ridership always increases by about 5,000 riders every December, that's an additive pattern.

Multiplicative seasonality assumes the seasonal swings scale with the level of the series. The seasonal component is multiplied by the level and trend. Use this when the peaks and troughs grow larger as the series rises. For example, if a retailer's holiday sales spike is always about 30% above the baseline, the absolute size of that spike grows as overall sales grow. That's multiplicative.

How to decide: Plot the raw series. If the seasonal "waves" stay the same height across the plot, go additive. If they fan out (get taller as the series rises), go multiplicative. Domain knowledge helps too: percentage-based effects tend to be multiplicative, while fixed-amount effects tend to be additive.

Smoothing parameters in Holt-Winters'

The three parameters α , β , and γ each sit on a spectrum:

- Values closer to 1 make the model react quickly to recent changes. This is useful when the underlying pattern shifts frequently, but it can also make forecasts noisy.
- Values closer to 0 produce smoother, more stable estimates. The model "remembers" older observations longer, which helps when the pattern is stable but hurts when it changes.

How parameters are estimated:

1. Grid search — try many combinations of α , β , and γ (e.g., each from 0.01 to 0.99 in small increments) and pick the combination that minimizes an error metric like MSE.
2. Numerical optimization — algorithms like L-BFGS or Nelder-Mead search the parameter space more efficiently to minimize the same error metric.

In either case, you should use cross-validation (often time-series-specific approaches like rolling-origin evaluation) to make sure the chosen parameters generalize well and don't just overfit the training data.

Forecasting with Holt-Winters'

The forecast equations combine the three components to project future values:

- Additive: $\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$
- Multiplicative: $\hat{y}_{t+h|t} = (\ell_t + hb_t) \times s_{t+h-m(k+1)}$

Where:

- $\hat{y}_{t+h|t}$ = forecast for h periods ahead, using information up to time t
- ℓ_t = estimated level at time t
- b_t = estimated trend (slope) at time t
- $s_{t+h-m(k+1)}$ = seasonal factor for the relevant season
- m = number of seasons per cycle (e.g., 12 for monthly data, 4 for quarterly)
- k = integer part of $(h-1)/m$, which ensures the model pulls the correct seasonal factor from the most recent complete cycle

Notice the structural difference: in the additive version, seasonality is added as a fixed amount; in the multiplicative version, it scales the level-plus-trend quantity.

Evaluating forecast accuracy:

1. MAE (Mean Absolute Error) — average of $|y_t - \hat{y}_t|$. Easy to interpret in the original units.
2. MSE (Mean Squared Error) — average of $(y_t - \hat{y}_t)^2$. Penalizes large errors more heavily.
3. MAPE (Mean Absolute Percentage Error) — average of $|y_t - \hat{y}_t|/|y_t|$, expressed as a percentage. Useful for comparing across series with different scales, but can be misleading when actual values are near zero.

Comparing these metrics against alternative methods (e.g., ARIMA, simpler exponential smoothing) tells you whether the added complexity of Holt-Winters' is actually paying off for your data.

Applications of Holt-Winters'

Holt-Winters' is a strong choice whenever your data shows both a trend and a clear seasonal cycle. Common examples include:

- Retail sales — monthly clothing sales that peak in November/December
- Tourism demand — quarterly hotel occupancy that spikes in summer
- Energy consumption — daily or monthly electricity demand with winter heating and summer cooling peaks

Typical workflow:

1. Preprocess — handle missing values (e.g., linear interpolation) and outliers. Confirm the frequency of your data (daily, monthly, quarterly).
2. Visualize — plot the series to check for trend and seasonality, and to judge whether the seasonal pattern looks additive or multiplicative.
3. Select the model — choose additive or multiplicative Holt-Winters' based on your visual inspection and domain knowledge.
4. Estimate parameters — use grid search or optimization to find α , β , and γ , validating with a hold-out set or rolling-origin cross-validation.
5. Forecast and evaluate — generate predictions for future periods and assess accuracy with MAE, MSE, or MAPE.
6. Interpret — communicate results in context. Be upfront about limitations: Holt-Winters' assumes the seasonal pattern repeats with a fixed period and may struggle with abrupt structural breaks or irregular seasonality.

Unit 6 – ARIMA Models

6.1 Autoregressive (AR) models

Introduction to Autoregressive (AR) Models

Autoregressive (AR) models predict the current value of a time series using its own past values. They're one of the foundational building blocks of ARIMA modeling, and understanding them well makes the rest of Unit 6 much easier to follow.

The core idea is simple: what happened recently in a time series often tells you something about what's happening now. AR models formalize that intuition by assuming a linear relationship between the current observation and some number of previous observations.

Key characteristics of AR models

- The time series must be stationary, meaning its mean, variance, and autocorrelation structure don't change over time.
- They capture autocorrelation, which is the correlation between an observation and its own lagged values.
- The relationship between current and past values is assumed to be linear.
- The goal is to describe the series's dynamics with as few parameters as possible.

Order of AR models

The order of an AR model, written as AR(p), tells you how many lagged terms the model uses. An AR(1) model uses only the immediately previous value; an AR(2) model uses the two most recent values, and so on.

The general equation for an AR(p) model:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + \varepsilon_t$$

Where:

- y_t is the current observation at time t
- c is a constant term (related to the series mean)
- $\phi_1, \phi_2, \dots, \phi_p$ are the autoregressive coefficients
- $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ are the lagged observations
- ε_t is the error term (white noise) at time t

How to choose p: You determine the order by looking at the partial autocorrelation function (PACF) plot. The PACF shows the correlation between y_t and y_{t-k} after removing the effects of all intermediate lags. If the PACF has significant spikes at lags 1 and 2 but cuts off after that, an AR(2) model is a reasonable choice.

Parameter Estimation and Model Assessment

Parameter estimation for AR models

Once you've chosen the order, you need to estimate the coefficients ($\phi_1, \phi_2, \dots$) and the constant C . Two main methods are used:

Ordinary Least Squares (OLS): - Minimizes the sum of squared residuals between observed and predicted values. - Computationally straightforward since it reduces to solving a system of linear equations. - Provides unbiased estimates when the errors are uncorrelated and have constant variance.

Maximum Likelihood Estimation (MLE): - Finds the parameter values that make the observed data most probable under the model. - More flexible than OLS because it can handle more complex error structures. - Requires numerical optimization, so it's more computationally intensive.

For most intro-level AR modeling with well-behaved data, OLS and MLE produce very similar results. OLS is the simpler starting point; MLE becomes more useful when error assumptions get more complicated.

Stationarity assessment of AR models

An AR model only makes sense if the series is stationary. Checking stationarity comes down to examining the autoregressive coefficients.

For an AR(1) model $y_t = C + \phi_1 y_{t-1} + \varepsilon_t$:

- Stationary if $|\phi_1| < 1$. The effect of past shocks gradually fades.
- Random walk if $\phi_1 = 1$. Shocks persist forever, and the series drifts without reverting to a mean.
- Explosive if $|\phi_1| > 1$. Values grow without bound, which isn't realistic for forecasting.

For higher-order AR(p) models, you check the characteristic equation:

$$1 - \phi_1 z - \phi_2 z^2 - \dots - \phi_p z^p = 0$$

The model is stationary if all roots of this equation lie outside the unit circle in the complex plane. You don't typically solve this by hand; statistical software handles it for you.

If your model turns out to be non-stationary, the standard fix is differencing the series (subtracting each value from the previous one) to induce stationarity before fitting the AR model. This is exactly where the "I" in ARIMA comes from.

Application and Interpretation

Applying AR models step by step

1. Identify the order using the PACF plot, or use model selection criteria like AIC (Akaike Information Criterion) or BIC (Bayesian Information Criterion) to compare candidate models.
2. Estimate the parameters using OLS or MLE.
3. Assess the fit through diagnostic checks: examine residuals for remaining autocorrelation, check for normality, and look for patterns the model missed.
4. Forecast future values using the fitted model.

Interpreting the coefficients

- A positive ϕ coefficient means the current value tends to move in the same direction as the lagged value. A negative coefficient means they tend to move in opposite directions.
- Larger magnitudes (closer to 1) mean past values have a stronger, more persistent influence on the present. Smaller magnitudes mean the effect fades quickly.
- The constant C is related to the overall mean of the series. When all lagged terms are at zero, C represents the baseline level.

Common applications

AR models are widely used for forecasting stock prices, exchange rates, and economic indicators. They also appear in energy demand modeling, sales forecasting, and web traffic analysis.

That said, AR models have real limitations. They assume linearity and rely solely on the series's own past. If external factors (like policy changes or weather events) drive the series, you may need to incorporate exogenous variables or use a more flexible model.

6.2 Moving average (MA) models

Moving Average (MA) Models

Moving Average (MA) models describe how an observation depends on past forecast errors (also called residuals or shocks). Unlike autoregressive models, which relate an observation to its own past values, MA models relate it to past mistakes in prediction. This makes them especially useful for capturing short-term dependencies in time series data.

MA models are always stationary: their mean, variance, and autocovariance don't change over time. A key property is that shocks have a finite memory. A past error only influences the series for a limited number of time steps, then its effect disappears entirely.

Moving Average Models in Time Series

An MA model says: "The current value equals the series mean, plus the current error, plus some weighted combination of recent past errors."

- Forecast errors (ε_t) are the differences between what actually happened and what the model predicted. These are assumed to be white noise (independent, identically distributed with mean zero).
- Because the model only looks back a fixed number of steps, the influence of any single shock eventually drops to zero. This is what "finite memory" means.
- MA models are inherently stationary. You don't need to difference the data or check for unit roots to ensure stationarity when working with a pure MA process.

Order of MA Models

The order of an MA model, written as $MA(q)$, tells you how many past error terms the model uses. An $MA(1)$ uses only the immediately previous error; an $MA(2)$ uses the two most recent errors, and so on.

The general equation for an $MA(q)$ model is:

$$y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q}$$

Where:

- y_t = observed value at time t
- μ = mean of the time series
- ε_t = white noise error term at time t
- $\theta_1, \theta_2, \dots, \theta_q$ = moving average coefficients (these are the parameters you estimate)

For a concrete example, an $MA(1)$ model looks like $y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1}$. The current value depends on the mean, the current shock, and one past shock weighted by θ_1 .

Parameter Estimation for MA Models

Estimating MA parameters is trickier than for AR models because the error terms ε_t are not directly observed. Two main approaches are used:

Least Squares Estimation - Minimizes the sum of squared residuals between observed and predicted values. - Provides estimates for the moving average coefficients (θ) and the error variance. - Because past errors aren't observed, this typically requires an iterative procedure where initial error values are assumed (often set to zero) and then refined.

Maximum Likelihood Estimation (MLE) - Finds the parameter values that make the observed data most probable under the assumed model. - Also iterative, but generally produces more efficient estimates (lower variance) than least squares, especially for shorter series. - MLE is the default in most statistical software packages for fitting MA models.

The choice between methods matters less as your sample size grows. For short series, MLE tends to perform better.

Invertible vs. Non-Invertible MA Models

Invertibility is a condition that guarantees a unique MA model representation for a given autocorrelation structure. Without it, multiple different sets of parameters could produce the same autocorrelation pattern, which creates ambiguity.

An invertible MA model can be rewritten as an infinite-order AR model. This is the MA analog of stationarity for AR models: just as we require AR models to be stationary, we require MA models to be invertible.

Invertibility conditions for $MA(q)$:

- The roots of the characteristic polynomial $1 + \theta_1 z + \theta_2 z^2 + \dots + \theta_q z^q = 0$ must all lie outside the unit circle (i.e., have absolute value greater than 1).
- For the simple $MA(1)$ case, this reduces to $|\theta_1| < 1$.

When invertibility fails:

- Multiple parameter sets can generate the same ACF, so you can't uniquely identify the model.
- Forecasting becomes unreliable because the mapping between errors and observations is ambiguous.
- In practice, most software will estimate the invertible version by default, but it's worth checking.

Application of MA Models to Data

Fitting an MA model follows a structured workflow:

1. Identify the model order by examining the autocorrelation function (ACF) and partial autocorrelation function (PACF). For a pure $MA(q)$ process, the ACF cuts off sharply after lag q (drops to zero), while the PACF decays gradually. This sharp cutoff in the ACF is the signature clue that you're dealing with an MA process.
2. Estimate parameters using MLE or least squares.
3. Check model fit with diagnostic tools: plot the residuals to verify they look like white noise, run a Ljung-Box test for remaining autocorrelation, and compare models using AIC or BIC (lower is better).
4. Forecast using the fitted model and interpret results in context.

Interpreting the coefficients:

- Each θ_i represents how much a past forecast error at lag i influences the current observation.
- A positive θ_1 means that when the model under-predicted in the previous period (positive error), the current value tends to be pushed higher as well.
- A negative θ_1 means past positive errors tend to pull the current value down, creating a corrective or oscillating pattern.
- Larger absolute values of θ indicate stronger influence from that lag's error on the present observation.

6.3 Mixed ARMA models

Introduction to Mixed ARMA Models

ARMA models combine autoregressive (AR) and moving average (MA) components into a single framework for modeling stationary time series. Where a pure AR model only uses past values and a pure MA model only uses past errors, an ARMA model uses both, making it more flexible and often more parsimonious (fewer parameters needed to capture the same patterns). Getting comfortable with ARMA is essential before moving on to ARIMA, which adds differencing for non-stationary data.

Characteristics of ARMA Models

The AR component models how the current value depends on its own past values. The MA component models how the current value depends on past forecast errors (white noise terms). Together, they can capture both the momentum of a series and the short-lived effects of random shocks.

A core assumption is stationarity: the mean, variance, and autocovariance of the series don't change over time. If your data has a trend or changing variance, you need to transform it before fitting an ARMA model.

The general ARMA(p, q) model is written as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

Where:

- y_t is the observed value at time t
- c is a constant (related to the mean of the series)

- $\phi_1, \phi_2, \dots, \phi_p$ are the autoregressive coefficients
- $\theta_1, \theta_2, \dots, \theta_q$ are the moving average coefficients
- $\varepsilon_t, \varepsilon_{t-1}, \dots, \varepsilon_{t-q}$ are white noise error terms (mean zero, constant variance, uncorrelated)

Orders in ARMA Models

The notation ARMA(p, q) tells you exactly how many terms of each type the model includes:

- p = number of lagged observations (AR order)
- q = number of lagged error terms (MA order)

For example, an ARMA(1, 1) model looks like:

$$y_t = c + \phi_1 y_{t-1} + \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

This is the simplest mixed model: one past value and one past error. An ARMA(2, 1) would add a second lagged observation ($\phi_2 y_{t-2}$) while keeping just one lagged error.

Notice that ARMA(p, 0) is just a pure AR(p) model, and ARMA(0, q) is a pure MA(q) model. So ARMA is really a generalization of both.

Choosing p and q relies on examining two diagnostic plots:

- ACF (autocorrelation function): measures correlation between observations at different lags. For a pure AR process, the ACF decays gradually; for a pure MA process, it cuts off sharply after lag q.
- PACF (partial autocorrelation function): measures correlation at each lag after removing the effects of shorter lags. For a pure AR process, the PACF cuts off after lag p; for a pure MA process, it decays gradually.

When both the ACF and PACF show gradual decay rather than a clean cutoff, that's a strong signal you need a mixed ARMA model rather than a pure AR or MA.

Estimation and Assessment of ARMA Models

Parameter Estimation

Once you've chosen p and q, you need to estimate the coefficients. Two main approaches:

- Least squares estimation finds the values of ϕ_i , θ_j , and c that minimize the sum of squared residuals (the differences between observed and predicted values). This is conceptually straightforward but can be computationally tricky for the MA part, since past errors aren't directly observed.
- Maximum likelihood estimation (MLE) finds the parameter values that make the observed data most probable under the assumed model. MLE is generally preferred because it has better statistical properties (efficiency, consistency) and handles the MA component more naturally.

In practice, most software uses MLE or a conditional least squares approach by default.

Stationarity and Invertibility

These two conditions determine whether your fitted model is valid and well-behaved.

Stationarity (applies to the AR part): The model is stationary if the roots of the AR characteristic polynomial lie outside the unit circle.

$$1 - \phi_1 z - \phi_2 z^2 - \dots - \phi_p z^p = 0$$

If all roots of this equation satisfy $|z| > 1$, the model is stationary. For the simple AR(1) case, this reduces to $|\phi_1| < 1$, which is easier to check by hand.

Invertibility (applies to the MA part): The model is invertible if the roots of the MA characteristic polynomial lie outside the unit circle.

$$1 + \theta_1 z + \theta_2 z^2 + \dots + \theta_q z^q = 0$$

Invertibility guarantees a unique MA representation and ensures that recent observations carry more weight than distant ones. For an MA(1), this simplifies to $|\theta_1| < 1$.

A model that fails either condition can produce unreliable forecasts and misleading coefficient estimates. If you encounter this, reconsider your choice of p and q or check whether the data truly is stationary.

Applying ARMA Models in Practice

Fitting an ARMA model follows a structured workflow:

1. Select orders (p, q) by examining the ACF and PACF plots of your stationary series
2. Estimate parameters using MLE or least squares
3. Check model fit with diagnostic tools: - Residual analysis: residuals should look like white noise (no remaining autocorrelation, roughly normal distribution) - Information criteria: AIC and BIC penalize model complexity, so lower values indicate a better balance of fit and simplicity. If ARMA(1,1) and ARMA(2,1) fit similarly, the information criteria help you pick the simpler one
4. Interpret the coefficients: - AR coefficients (ϕ) reflect how strongly past values persist into the present. A ϕ_1 of 0.8 means the series has strong momentum. - MA coefficients (θ) reflect how past shocks ripple forward. A θ_1 of -0.5 means a positive shock at time $t - 1$ pulls the current value down by half that shock's size.
5. Generate forecasts from the fitted model and evaluate accuracy against held-out data
6. Acknowledge limitations: ARMA models assume linearity and stationarity. Real-world data often violates these assumptions, which is why ARIMA (adding differencing) and other extensions exist

6.4 Integrated ARIMA models

ARIMA models are powerful tools for analyzing time series data, combining autoregressive, differencing, and moving average components. They can handle both stationary and non-stationary data, making them versatile for various applications like stock prices and weather forecasting.

Understanding the orders (p,d,q) in ARIMA models is crucial. The AR order (p) relates current observations to past ones, differencing order (d) achieves stationarity, and MA order (q) incorporates past error terms. These components work together to capture complex patterns in time series data.

Integrated ARIMA Models

Definition of ARIMA models

- ARIMA models combine autoregressive (AR), differencing (I), and moving average (MA) components to analyze time series data

- AR component captures the relationship between an observation and a certain number of lagged observations (previous values in the series)
- I component involves differencing the time series to achieve stationarity (constant mean, variance, and autocovariance over time)
- MA component models the error term as a linear combination of error terms occurring at various times in the past (residuals)
- ARIMA models are denoted as $ARIMA(p,d,q)$
- p is the order of the AR term (number of lagged observations)
- d is the degree of differencing (number of times the series is differenced)
- q is the order of the MA term (number of lagged error terms)
- ARIMA models handle both stationary and non-stationary time series data
- Useful for modeling time series with trends, seasonal patterns, and short-term dependencies (stock prices, weather data)

Orders in ARIMA models

- The order of an ARIMA model is determined by three parameters: p , d , and q
- p represents the number of autoregressive terms in the model
- An $AR(p)$ model relates the current observation to the past p observations
- $AR(1)$ model relates the current observation to the immediately preceding observation (lag of 1)
- d represents the number of differencing operations applied to the time series to achieve stationarity
- Differencing involves computing the differences between consecutive observations
- A time series that requires d differencing operations to become stationary is integrated of order d , denoted as $I(d)$
- q represents the number of moving average terms in the model
- An $MA(q)$ model relates the current observation to the past q error terms
- $MA(1)$ model relates the current observation to the immediately preceding error term (lag of 1)

Stationarity and Parameter Estimation

Differencing for stationarity

- Stationarity is a crucial assumption for many time series models, including ARIMA
- A stationary time series has constant mean, variance, and autocovariance over time
- Differencing transforms a non-stationary time series into a stationary one
 1. Compute the first difference by subtracting each observation from its preceding observation
 2. If the first difference is not stationary, repeat the process to obtain higher-order differences
- The order of differencing (d) in an ARIMA model is determined by the number of differencing operations required to achieve stationarity

- Once stationarity is achieved, the differenced time series can be used for ARIMA modeling
- The original time series can be reconstructed by applying the inverse of the differencing operation to the model's predictions (cumulative sum)

Parameter estimation for ARIMA

- After determining the orders (p , d , q) of an ARIMA model, estimate the model parameters using appropriate statistical methods
- Least squares estimation minimizes the sum of squared residuals between the observed values and the model's predicted values
- Residuals are the differences between the observed values and the model's predictions
- Objective: find parameter values that minimize the sum of squared residuals
- Maximum likelihood estimation finds parameter values that maximize the likelihood function
- Likelihood function measures the probability of observing the given data under the assumed model
- Objective: find parameter values that make the observed data most likely
- Both methods aim to find the best-fitting parameter values for the ARIMA model
- Statistical software packages provide tools for estimating ARIMA model parameters (R, Python)

Interpretation of ARIMA results

- Interpreting ARIMA results involves understanding the impact of each component (AR, I, and MA) on the model's predictions and performance
- Autoregressive (AR) terms capture the relationship between an observation and its lagged values
- Positive AR coefficients: an increase in past values leads to an increase in the current value
- Negative AR coefficients: an increase in past values leads to a decrease in the current value
- Magnitude of AR coefficients determines the strength of the relationship between current and lagged values
- Differencing (I) component removes trends and seasonality from the time series
- Interpreting results of a differenced series requires considering the order of differencing and its impact on the original time series
- Higher orders of differencing may remove more complex patterns but can make the series more difficult to interpret
- Moving average (MA) terms capture the relationship between an observation and past forecast errors
- Positive MA coefficients: a positive error in the past leads to an increase in the current value
- Negative MA coefficients: a positive error in the past leads to a decrease in the current value
- Magnitude of MA coefficients determines the strength of the relationship between current value and past errors
- Assess overall performance of an ARIMA model using evaluation metrics

- Mean squared error (MSE), root mean squared error (RMSE), or mean absolute error (MAE)
- Lower values of these metrics indicate better model performance
- Interpret ARIMA results in the context of the problem domain, considering practical implications of the model's predictions and limitations of the modeling approach

Unit 7 – Model Selection & Diagnostic Checking

7.1 Information criteria (AIC, BIC) for model selection

Model Selection in Time Series Analysis

Time series model selection is about finding the sweet spot between fit and complexity. A model that's too simple misses real patterns in your data (underfitting), while one that's too complex starts fitting random noise (overfitting). The Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) are the two main tools for navigating this tradeoff.

Importance of Model Selection

Time series data often contain layered patterns: autocorrelation, seasonality, and trend all at once. Model selection is the process of choosing which candidate model best captures those patterns without becoming unnecessarily complex.

- Overfitting happens when a model has too many parameters and starts fitting noise rather than signal. It looks great on your training data but forecasts poorly.
- Underfitting happens when a model is too simple to capture the real structure in the data.
- The goal is a model that generalizes well, meaning it performs accurately on new, unseen data, not just the data it was trained on.

Role of the Akaike Information Criterion (AIC)

AIC was developed by Hirotugu Akaike in 1974. It evaluates a model by weighing how well it fits the data against how many parameters it uses.

The formula is:

$$AIC = 2k - 2\ln(L)$$

where k is the number of estimated parameters and L is the maximized likelihood of the model (a measure of how probable the observed data are under that model).

A few things to note about AIC:

- Lower AIC = better model. The $-2\ln(L)$ term rewards good fit, while the $2k$ term adds a penalty for each additional parameter.
- The penalty per parameter is fixed at 2, regardless of sample size. This means AIC can sometimes favor slightly more complex models, especially with large datasets.
- AIC allows you to compare non-nested models, meaning models that aren't simply restricted versions of each other (e.g., comparing an ARIMA model to an exponential smoothing model).

Bayesian Information Criterion (BIC) vs. AIC

BIC was developed by Gideon Schwarz in 1978. It takes a similar approach but incorporates the sample size into its complexity penalty.

The formula is:

$$BIC = k\ln(n) - 2\ln(L)$$

where k is the number of parameters, n is the sample size, and L is the maximized likelihood.

The key difference is in the penalty term. AIC penalizes each parameter by 2, while BIC penalizes each parameter by $\ln(n)$. Since $\ln(n) > 2$ whenever $n > 7$ (which is almost always in time series work), BIC penalizes complexity more heavily than AIC for any reasonably sized dataset.

AIC tends to select slightly more complex models. It's often preferred when the goal is short-term forecast accuracy.

BIC tends to select simpler, more parsimonious models. It has a theoretical property called consistency: as sample size grows, BIC will select the true model (if it's among the candidates) with probability approaching 1. AIC does not have this property.

In practice, AIC and BIC often agree. When they disagree, it's usually because AIC picks a model with one or two extra parameters. Neither is universally "better"; the right choice depends on your analysis goals.

Applying AIC and BIC

Here's the typical workflow:

1. Fit candidate models to your time series data (e.g., several ARIMA orders, SARIMA variants, or exponential smoothing models).
2. Calculate the log-likelihood and count the parameters for each fitted model. Most software does this automatically.
3. Compute AIC and BIC for each model using the formulas above.
4. Select the model with the lowest AIC or BIC value as your top candidate.

A couple of important caveats:

- AIC and BIC give you a relative ranking of your candidate models. A model with the lowest AIC in your set could still be a poor model overall if none of your candidates are appropriate. Always check residual diagnostics (covered later in this unit) to confirm the selected model is adequate.
- Think about the purpose of your analysis. If you need the best short-term forecasts, AIC's slight preference for complexity may help. If you want to identify the simplest model that explains the data's structure, BIC is often the better guide.

7.2 Residual analysis and diagnostic tests

Residual Analysis and Diagnostic Tests

Residual analysis is how you check whether your time series model is actually doing its job. After fitting a model, the residuals (the differences between what you observed and what the model predicted) should look like random noise. If they don't, your model is missing something. This section covers the main plots and statistical tests you'll use to diagnose problems and decide how to fix them.

Importance of Residual Analysis

A well-specified time series model should extract all the systematic patterns from the data, leaving behind only unpredictable noise. Residual analysis tests whether that's actually happened.

- Residuals are the differences between observed values and model-fitted values: $e_t = y_t - \hat{y}_t$
- If the model is good, residuals should behave like white noise: no patterns, no autocorrelation, roughly constant variance, and approximately normal distribution
- Residual analysis also helps you spot outliers or influential observations that may be distorting your model's estimates

Think of it this way: if you can still find a pattern in the residuals, that pattern is information your model failed to capture, and you should go back and improve the model.

Interpretation of Residual Plots

Residual plots are your first line of defense. Before running any formal tests, visually inspect these graphs:

- Residuals vs. fitted values: checks for heteroscedasticity (non-constant variance) and non-linearity
- Ideal: residuals scattered randomly around zero with no fan shape or curve
- Residuals vs. time: checks for trends, seasonal patterns, or shifts in variance over time
- Ideal: residuals fluctuate randomly around zero with no systematic drift
- ACF and PACF of residuals: checks for leftover autocorrelation the model didn't capture
- Ideal: nearly all spikes fall within the 95% confidence bands

When you spot a problem in these plots, it points toward a specific fix:

Pattern You See	What It Suggests	Possible Fix
Fan or funnel shape in residuals vs. fitted	Non-constant variance (heteroscedasticity)	Variance-stabilizing transformation (e.g., log) or a GARCH-type model
Curved pattern in residuals vs. fitted	Non-linearity	Add non-linear terms or apply a transformation
Trend or drift in residuals vs. time	Non-stationarity not fully addressed	Additional differencing or trend terms
Significant spikes in ACF/PACF	Residual autocorrelation	Add AR or MA terms to the model

Diagnostic Tests for Autocorrelation

Visual inspection of ACF/PACF plots is useful but subjective. Formal tests give you a more rigorous answer about whether residual autocorrelation is present.

Ljung-Box Test

This is the most commonly used test for residual autocorrelation in time series. It checks multiple lags simultaneously rather than testing one lag at a time.

- Null hypothesis (H_0): Residuals are independently distributed (no autocorrelation up to lag h)
- Alternative hypothesis (H_a): Residuals exhibit autocorrelation at one or more lags

The test statistic is:

$$Q = n(n+2) \sum_{k=1}^h \frac{r_k^2}{n-k}$$

where n is the sample size, h is the number of lags tested, and r_k is the sample autocorrelation of residuals at lag k .

How to use it:

1. Choose the number of lags h to test (a common rule of thumb is $h = 10$ for non-seasonal data, or $h = 2m$ where m is the seasonal period)
2. Compute the Q statistic from the residual autocorrelations
3. Compare Q to the critical value from a chi-square distribution with $h - p - q$ degrees of freedom, where p and q are the number of AR and MA parameters in your model
4. If Q exceeds the critical value (or equivalently, the p-value is below your significance level), reject H_0 and conclude that significant autocorrelation remains

Note on degrees of freedom: Some textbooks use h degrees of freedom directly, but when testing residuals from an ARMA(p, q) model, you should subtract the number of estimated parameters ($h - p - q$) to account for the fact that the residuals aren't truly independent observations.

Other autocorrelation tests:

- Durbin-Watson test: tests specifically for first-order (lag-1) autocorrelation; limited because it only checks one lag
- Breusch-Godfrey test: more flexible than Durbin-Watson, can test for autocorrelation at multiple lags and works even when lagged dependent variables are present

If any of these tests detect significant autocorrelation, your model is misspecified. The typical remedy is adding AR or MA terms to capture the remaining dependence.

Assessment of Residual Normality

Many time series inference procedures (confidence intervals, prediction intervals, hypothesis tests on coefficients) assume the errors are normally distributed. Residual normality checks tell you whether those results are trustworthy.

Start with a visual check: a histogram of residuals and a Q-Q plot (quantile-quantile plot comparing residual quantiles to theoretical normal quantiles). If the Q-Q plot shows points roughly along a straight line, normality is reasonable.

Jarque-Bera Test

This test checks normality by looking at whether the residuals have the skewness and kurtosis you'd expect from a normal distribution (skewness = 0, kurtosis = 3).

- Null hypothesis (H_0): Residuals are normally distributed
- Alternative hypothesis (H_a): Residuals are not normally distributed

The test statistic is:

$$JB = \frac{n}{6} \left(S^2 + \frac{(K-3)^2}{4} \right)$$

where n is the sample size, S is the sample skewness, and K is the sample kurtosis. Under H_0 , this follows a chi-square distribution with 2 degrees of freedom.

A large JB value (small p-value) means the residuals deviate significantly from normality, either through asymmetry (skewness) or heavy/light tails (kurtosis).

Other normality tests include the Shapiro-Wilk test (generally more powerful for small samples) and the Anderson-Darling test.

If normality is violated, you have a few options:

1. Transform the data (e.g., log or Box-Cox transformation) before fitting the model, which often stabilizes variance and improves normality simultaneously
2. Use robust estimation methods that are less sensitive to non-normal errors
3. Use bootstrap or distribution-free methods for inference instead of relying on normal-theory confidence intervals

Keep in mind that mild departures from normality are usually not a serious problem, especially with larger samples, because many estimators are still consistent. Severe skewness or heavy tails are more concerning and worth addressing.

7.3 Overfitting and underfitting

Time series models can be too simple or too complex, leading to underfitting or overfitting. Underfitting occurs when models miss key patterns, while overfitting captures noise, causing poor predictions. Balancing complexity is crucial for accurate forecasting.

Techniques like cross-validation, regularization, and early stopping help prevent overfitting. These methods ensure models generalize well to new data, striking a balance between fitting historical patterns and making reliable future predictions in time series analysis.

Model Complexity and Generalization

Definition of overfitting vs underfitting

- Overfitting happens when a model becomes too complex, capturing noise and random fluctuations in the training data (stock price fluctuations) rather than the underlying patterns
- Model learns the idiosyncrasies and outliers of the training data (unusual spikes or dips in stock prices) instead of generalizing well to unseen data
- Leads to poor performance and unreliable predictions on new, unseen data (future stock prices)
- Underfitting occurs when a model is too simple, failing to capture the relevant patterns and relationships in the data (long-term trends in stock prices)
- Model is not complex enough to learn the important information from the training data (seasonal patterns or market cycles)
- Results in high bias, causing the model to perform poorly on both the training data and new, unseen data (consistently inaccurate stock price predictions)

Consequences of model fit issues

- Overfitting consequences
 - Model achieves excellent performance on the training data (low error on historical stock prices) but performs poorly on new, unseen data (inaccurate future stock price predictions)
- Overfitted models have high variance and low bias, meaning they are sensitive to small fluctuations in the training data (reacting to random noise in stock prices)
- Leads to poor generalization and unreliable predictions on future data (stock price forecasts that don't hold up in reality)

- Underfitting consequences
- Model performs poorly on both the training data and new, unseen data (inaccurate stock price predictions across the board)
- Underfitted models have high bias and low variance, consistently making wrong assumptions and failing to capture the underlying patterns (missing important market trends or company-specific events)
- Fails to capture the true relationships and patterns in the data (inability to model the factors driving stock prices)

Complexity vs generalization trade-off

- Model complexity refers to the number of parameters and the flexibility of the model (number of variables and interactions considered in a stock price prediction model)
- Increasing model complexity can lead to better fit on the training data (lower error on historical stock prices)
- However, overly complex models are prone to overfitting (considering too many variables and noise, leading to poor future predictions)
- Generalization ability refers to the model's performance on new, unseen data (accuracy of stock price predictions on future, out-of-sample data)
- A model with good generalization can accurately predict future observations (reliable stock price forecasts)
- Striking the right balance between model complexity and generalization is key for optimal performance (a stock price model that fits historical data well but also generalizes to future data)

Techniques for mitigating overfitting

- Cross-validation assesses the model's performance on unseen data (testing a stock price model on data from different time periods)
- Involves splitting the data into multiple subsets (k-fold cross-validation on stock price data)
 1. Data is divided into k equal-sized subsets (e.g., 5-fold cross-validation on 5 years of stock price data)
 2. Model is trained on k-1 subsets and evaluated on the remaining subset (training on 4 years and testing on the 5th year)
 3. Process is repeated k times, with each subset used once for evaluation (rotating the test year)
 4. Performance is averaged across all k iterations to get an estimate of the model's performance on unseen data (average prediction accuracy across all test years)
- Helps detect overfitting by providing an estimate of the model's performance on new data (identifying if the model is overfitting to specific time periods)
- Regularization techniques penalize complex models to mitigate overfitting (adding a penalty term to the model's loss function)
- L1 regularization (Lasso) adds a penalty proportional to the absolute value of the model's coefficients, encouraging sparse models (setting less important variables to zero)

- L2 regularization (Ridge) adds a penalty proportional to the square of the model's coefficients, shrinking them towards zero (reducing the impact of less important variables)
- Early stopping halts the training process when the model's performance on a validation set starts to degrade (stopping training when the error on unseen stock price data starts increasing)
- Simplifying the model architecture (reducing the number of layers or neurons in a neural network) or reducing the number of features (selecting only the most relevant variables for stock price prediction) can also help mitigate overfitting

Unit 8 – Forecasting with Time Series Models

8.1 Point forecasts and prediction intervals

Point Forecasts and Prediction Intervals

Point Forecasts in Time Series

A point forecast is a single-number estimate of a future value in a time series. Think of it as your best guess for what will happen at a specific future time point, like predicting next quarter's sales will be 50,000 units.

You generate point forecasts using methods like exponential smoothing or ARIMA models (for example, Holt-Winters or ARIMA(1,1,1)). These concrete estimates drive practical decisions: how much inventory to stock, how many staff to schedule, how to allocate a budget.

Point forecasts also serve two other purposes:

- They give you a basis for comparing models. You can evaluate which forecasting method performs best using error metrics like mean squared error (MSE) or mean absolute percentage error (MAPE).
- They act as the center point around which you build prediction intervals.

Concept of Prediction Intervals

A point forecast alone can be misleading because it hides how confident (or uncertain) you actually are. A prediction interval fixes this by providing a range of values likely to contain the true future observation at a specified probability level.

For example, you might report: "We forecast 125 units, with a 95% prediction interval of [100, 150]." That means you're 95% confident the actual value will fall between 100 and 150.

Key properties of prediction intervals:

- They consist of a lower bound and an upper bound centered on the point forecast.
- The probability level you choose (80%, 90%, 95%) determines how wide the interval is. Higher confidence means a wider interval.
- Wider intervals signal greater uncertainty, while narrower intervals suggest a more precise forecast. Short-term forecasts tend to have narrower intervals than long-term ones, since uncertainty grows over time.

Construction of Prediction Intervals

There are two broad approaches to building prediction intervals: parametric and non-parametric.

Parametric methods assume the forecast errors follow a known distribution, most commonly a normal distribution. Under that assumption, you construct the interval using standard normal quantiles:

$$\hat{y}_{T+h} \pm z_{\alpha/2} \times \sigma_h$$

where $\hat{y}_{T+h}$ is the point forecast, $Z_{\alpha/2}$ is the critical value (e.g., 1.96 for a 95% interval), and σ_h is the standard deviation of the forecast errors at horizon h . The interval width is driven entirely by σ_h .

Non-parametric methods don't require distributional assumptions:

- Empirical quantiles: Sort your historical forecast errors and use the 2.5th and 97.5th percentiles directly (for a 95% interval).
- Bootstrap methods: Resample the model's residuals many times, generate many possible future paths, and read off the interval from the resulting distribution of forecasts.

Three factors that influence interval width:

1. Forecast horizon — Intervals almost always widen as you forecast further ahead. A one-month-ahead interval will be narrower than a one-year-ahead interval because uncertainty accumulates.
2. Model complexity — A well-specified complex model (like ARIMA) may produce narrower intervals than simple exponential smoothing because it captures more structure in the data. But beware of overfitting: a model that fits noise will produce intervals that are misleadingly narrow.
3. Variability in the series — A volatile series like daily stock prices will naturally produce wider intervals than a stable series like weekly demand for a household staple.

Uncertainty in Point Forecasts

Prediction intervals exist to remind you (and your stakeholders) that a point forecast is an estimate, not a guarantee. If your point forecast is 120 and your 95% prediction interval is [90, 150], you're saying the true value could plausibly land anywhere in that 60-unit range.

Interpreting prediction intervals carefully:

- The probability level you choose is a tradeoff. An 80% interval is narrower and more actionable but less likely to contain the true value. A 95% interval is safer but wider and potentially less useful for tight planning.
- Prediction intervals reflect uncertainty within your model. They do not account for structural breaks, regime shifts, or external shocks that your model has never seen. If the economy enters a recession and your model was trained on expansion-era data, the interval may be too narrow.
- Rare, extreme events (sometimes called black swan events) can fall outside even a 99% prediction interval. The intervals assume the future broadly resembles the past.

Communicating uncertainty to stakeholders:

- Always present prediction intervals alongside point forecasts, whether in a table or a fan chart. A single number without context invites overconfidence.
- Explain intervals in plain language: "There's roughly a 95% chance the actual value falls within this range."
- Connect uncertainty to decisions. If the lower bound of the interval would trigger a different action than the upper bound, that's a signal for contingency planning or scenario analysis.

8.2 Evaluating forecast accuracy (MAE, RMSE, MAPE)

Evaluating forecast accuracy tells you how well your model's predictions match reality. Without a reliable way to measure errors, you can't compare models, spot problems, or know whether your forecasts are actually useful for decisions like budgeting or inventory planning.

Three metrics show up constantly in practice: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). Each one captures forecast error differently, and knowing when to use which is just as important as knowing how to calculate them.

Evaluating Forecast Accuracy

Importance of forecast accuracy evaluation

Forecast accuracy evaluation isn't just a final checkbox. It's woven into every stage of building and maintaining a time series model.

- **Model selection:** Accuracy metrics help you pick the best model for your data (e.g., choosing between ARIMA and exponential smoothing based on which produces lower error on a holdout set).
- **Diagnosing problems:** Consistently high errors can reveal overfitting (model memorizes training data but fails on new data) or underfitting (model is too simple to capture the pattern).
- **Supporting decisions:** Forecasts drive real actions like staffing, purchasing, and budgeting. Knowing how wrong your forecasts tend to be lets decision-makers plan with appropriate caution.
- **Ongoing monitoring:** A model that worked well six months ago might degrade as underlying patterns shift. Tracking accuracy over time tells you when it's time to retrain or switch methods.

Mean Absolute Error calculation

MAE measures the average size of your forecast errors, ignoring their direction. It answers: on average, how far off are my predictions?

$$MAE = \frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t|$$

- n = number of forecast periods
- y_t = actual value at time t
- $\hat{y}_t$ = forecasted value at time t

How to compute it step by step:

1. For each time period, subtract the forecast from the actual value: $y_t - \hat{y}_t$.
2. Take the absolute value of each difference (drop the sign).
3. Sum all the absolute differences.
4. Divide by n (the number of observations).

Example: Suppose you forecast daily sales for 3 days. Actuals are 100, 150, 130 and forecasts are 110, 140, 135. The absolute errors are 10, 10, and 5. So $MAE = (10 + 10 + 5)/3 = 8.33$ units.

MAE is expressed in the same units as your data (dollars, units sold, etc.), which makes it straightforward to communicate. It's also less sensitive to outliers than RMSE because it doesn't square the errors. The

tradeoff is that MAE treats a 1-unit error and a 50-unit error with equal weight per unit of error, which may not reflect the true cost of large misses.

Root Mean Squared Error computation

RMSE also measures average error magnitude, but it penalizes larger errors more heavily because it squares each error before averaging.

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2}$$

- n = number of forecast periods
- y_t = actual value at time t
- $\hat{y}_t$ = forecasted value at time t

How to compute it step by step:

1. For each time period, calculate the error: $y_t - \hat{y}_t$.
2. Square each error.
3. Sum all the squared errors.
4. Divide by n .
5. Take the square root of the result.

Using the same example: Errors are -10, 10, -5. Squared errors are 100, 100, 25. Mean of squared errors = 75. $RMSE = \sqrt{75} \approx 8.66$ units.

Notice RMSE (8.66) is slightly higher than MAE (8.33) here. That gap widens when you have a few large errors mixed in with small ones, because squaring amplifies big misses. This property makes RMSE a better choice when large errors are especially costly (think financial risk or safety-critical applications). Like MAE, RMSE is in the same units as the data, but the squaring step makes it a bit less intuitive to explain to non-technical audiences.

Mean Absolute Percentage Error interpretation

MAPE converts errors into percentages of the actual values, making it scale-independent. This is useful when you need to compare accuracy across datasets measured in different units or magnitudes.

$$MAPE = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{y_t - \hat{y}_t}{y_t} \right|$$

- n = number of forecast periods
- y_t = actual value at time t
- $\hat{y}_t$ = forecasted value at time t

How to compute it step by step:

1. For each time period, calculate the error: $y_t - \hat{y}_t$.
2. Divide each error by the actual value y_t .
3. Take the absolute value of each percentage error.

4. Sum all the absolute percentage errors.

5. Divide by n and multiply by 100%.

Using the same example: Percentage errors are $|-10/100|$, $|10/150|$, $|-5/130| = 0.10, 0.067, 0.038$.
 $MAPE = 100\% \times (0.10 + 0.067 + 0.038)/3 \approx 6.83\%$.

A MAPE of 6.83% means your forecasts are off by about 6.83% on average. That's easy for stakeholders to grasp regardless of what the data measures.

The big limitation: MAPE divides by y_t , so if any actual value is zero (or very close to zero), the metric blows up or becomes misleading. Avoid MAPE for data that naturally includes zeros (e.g., intermittent demand).

Comparison of accuracy measures

	MAE	RMSE	MAPE
Units	Same as data	Same as data	Percentage
Outlier sensitivity	Lower	Higher (squaring amplifies large errors)	Depends on scale of actuals
Cross-scale comparison	Not ideal	Not ideal	Yes, scale-independent
Handles zeros in data	Yes	Yes	No (undefined when $y_t = 0$)
Interpretability	Very intuitive	Slightly less intuitive	Intuitive as a percentage
Optimization-friendly	Less so (absolute value not differentiable at 0)	Yes (quadratic, differentiable)	Less so

When to use which:

- MAE is your default go-to when you want a simple, robust summary of average error size and your data doesn't have extreme outlier concerns.
- RMSE is the better pick when large errors carry disproportionate costs, or when you're optimizing a model (many algorithms minimize squared error naturally).
- MAPE shines when you need to compare forecast quality across series with different scales (e.g., forecasting both a product that sells 50 units/day and one that sells 5,000 units/day), but only if your actuals stay well away from zero.

In practice, reporting more than one metric gives a fuller picture. If MAE and RMSE are close together, your errors are fairly uniform. If RMSE is much larger than MAE, a few big misses are pulling the average up, and that's worth investigating.

8.3 Cross-validation in time series

Cross-validation in time series forecasting is a crucial technique for assessing model performance. It helps determine how well a model can predict future values by using historical data, which is essential for making accurate forecasts.

This method involves partitioning data, training models on subsets, and validating on remaining data. It's particularly useful for comparing different models, tuning parameters, and ensuring models can generalize well to new, unseen data.

Cross-Validation in Time Series Forecasting

Concept of cross-validation

- Model validation technique assesses performance and generalization ability of predictive models
- Partitions available data into subsets, trains model on a subset, validates on remaining data
- Estimates how well model will perform on unseen data (out-of-sample performance)
- In time series forecasting, evaluates accuracy and robustness of forecasting models
- Assesses model's ability to capture underlying patterns and make accurate future predictions (next week, next month)
- Helps select best-performing model among different candidates (ARIMA, exponential smoothing)

Rolling-origin cross-validation application

- Rolling-origin cross-validation (time series cross-validation, walk-forward validation) tailored for time series data
- Preserves temporal order and dependencies in data
- Avoids using future information to predict past, preventing overly optimistic performance estimates
- Steps in rolling-origin cross-validation: 1. Define initial training set and validation set 2. Train model on initial training set 3. Evaluate model's performance on validation set 4. Expand training set by including next time point(s) from validation set 5. Repeat steps 2-4 until all available data points used for validation
- Performance metrics (mean squared error, mean absolute error) computed for each validation step, averaged for overall model performance estimate

Interpretation of cross-validation results

- Cross-validation results provide insights into model performance, aid in model selection
- Compare performance metrics of different models to identify one with lowest error or highest accuracy
- Consider stability and consistency of model's performance across different validation folds (robustness)
- Select model demonstrating best balance between performance and complexity
- Avoid overly complex models that may overfit data and perform poorly on unseen future data
- Prefer simpler models that generalize well and provide reliable forecasts (parsimony principle)

Advantages vs limitations of cross-validation

- Advantages of cross-validation in time series forecasting:
- Provides more reliable estimate of model's performance compared to single train-test split
- Allows assessment of model's robustness and stability over different time periods
- Helps select best model and tune its hyperparameters (lag order, smoothing parameters)

- Reduces risk of overfitting by evaluating model's performance on unseen data
- Limitations of cross-validation in time series forecasting:
- Assumes future data will have similar characteristics and patterns as historical data
- May not capture long-term trends or rare events not present in available data (structural breaks)
- Requires careful selection of validation set size and number of folds for meaningful results
- Can be computationally expensive, especially for large datasets and complex models (deep learning)

8.4 Combining forecasts

Combining Forecasts

Combining forecasts takes predictions from multiple models and merges them into a single, more reliable forecast. The core idea is straightforward: different models capture different patterns in your data, so blending them tends to outperform any single model on its own. This section covers why combination works, how to do it, and how to evaluate the result.

Rationale for Forecast Combination

No single time series model is perfect. An ARIMA model might capture trend and autocorrelation well but miss nonlinear patterns. Exponential smoothing might adapt quickly to level shifts but struggle with seasonal complexity. When you combine forecasts, you're hedging your bets across these strengths and weaknesses.

- Accuracy gains: Combined forecasts consistently outperform individual models in empirical studies, often by a meaningful margin.
- Reduced model risk: If one model is misspecified or fits noise rather than signal, the combination dilutes that error with better-performing models.
- Robustness: The combined forecast tends to be less sensitive to outliers, sudden spikes, or unusual patterns that might throw off a single model.

Simple Averaging of Forecasts

The most basic combination method is to take the arithmetic mean of all your model forecasts:

$$\text{Combined Forecast} = \frac{1}{n} \sum_{i=1}^n \text{Forecast}_i$$

where n is the number of models (e.g., ARIMA, exponential smoothing, a regression model).

This assigns equal weight to every model, assuming each contributes equally. It sounds almost too simple, but research has shown that simple averaging is surprisingly hard to beat, especially when you don't have a long track record to judge which model is best. It's also trivial to implement and easy to explain.

Weighted Averaging Techniques

When you have enough historical performance data, you can give better-performing models more influence in the combination. The general formula is:

$$\text{Combined Forecast} = \sum_{i=1}^n w_i \times \text{Forecast}_i$$

where the weights w_i sum to 1. The question is how to set those weights.

Inverse MSE Weighting

This approach rewards models that have produced smaller errors in the past:

1. Calculate MSE_i for each model i on a holdout or validation set.
2. Compute each weight as

$$w_i = \frac{1/MSE_i}{\sum_{j=1}^n 1/MSE_j}$$

3. Models with lower MSE get higher weights. For example, if Model A has an MSE of 4 and Model B has an MSE of 16, Model A receives 4 times the weight of Model B.

AIC Weighting

This method uses the Akaike Information Criterion, which balances fit and model complexity (parsimony):

1. Calculate AIC_i for each model i .
2. Compute each weight as

$$w_i = \frac{\exp(-0.5 \times AIC_i)}{\sum_{j=1}^n \exp(-0.5 \times AIC_j)}$$

3. Models with lower AIC (better fit relative to their complexity) receive higher weights.

AIC weighting is useful because it penalizes overly complex models, not just raw forecast error. However, both weighting schemes assume that past performance is a reasonable guide to future accuracy.

Accuracy of Combined vs. Individual Forecasts

After building a combined forecast, you need to check whether it actually improves on the individual models. Here's how to evaluate:

- Calculate standard error metrics for both the combined forecast and each individual model: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE).
- Compare across models. If the combined forecast has a lower MAE or MAPE than every individual model, the combination is adding value. If it doesn't, your weighting scheme may need adjustment, or one dominant model may be sufficient.
- Test across different periods. Check performance on training, validation, and test sets separately. A combined forecast that only wins on the training set but not the test set may be overfitting the weights.
- Consider the cost. Combining forecasts means maintaining and running multiple models, which takes more computational time and effort. If the accuracy gain is marginal, a single well-chosen model might be the more practical choice.

The general finding in the forecasting literature is that combinations almost always help, and simple averages are a strong default when you're unsure how to weight.

Unit 9 – Seasonal ARIMA Models

9.1 Seasonal differencing and SARIMA models

Seasonal Differencing and SARIMA Models

Many real-world time series have recurring patterns tied to a calendar cycle: retail sales spike every December, electricity demand peaks every summer, and so on. Seasonal differencing and SARIMA models give you the tools to handle these repeating patterns. Seasonal differencing removes the predictable seasonal component so the data becomes stationary, and SARIMA models then capture whatever structure remains in both the short-term and seasonal behavior.

Seasonal Differencing in Time Series

Regular differencing (subtracting consecutive observations) removes trends. Seasonal differencing does something analogous but across seasonal periods: it subtracts the observation from the same season in the previous cycle. If you have monthly data, you subtract the value from 12 months ago. If you have quarterly data, you subtract the value from 4 quarters ago.

The notation looks like this:

$$\nabla_s^D X_t = (1 - B^s)^D X_t$$

Here, s is the seasonal period (e.g., 12 for monthly data with an annual cycle), D is the order of seasonal differencing, and B^s is the backshift operator applied s times (so $B^{12}X_t = X_{t-12}$).

Why do this? Stationarity is a key assumption for ARIMA-type models. If the data has a seasonal component that shifts over time, the series isn't stationary. Seasonal differencing strips out that component, leaving behind trend and irregular parts that you can handle with regular differencing and ARMA terms.

The order D depends on how strong and persistent the seasonal pattern is. In practice, $D = 1$ handles the vast majority of cases. Higher orders are rare and can introduce unnecessary noise.

Structure of SARIMA Models

A SARIMA model extends ARIMA by adding a parallel set of seasonal parameters. The full notation is:

$$\text{SARIMA}(p, d, q)(P, D, Q)_s$$

The non-seasonal parameters work exactly like regular ARIMA: - P : autoregressive (AR) order - d : differencing order - q : moving average (MA) order

The seasonal parameters mirror them but operate at multiples of the seasonal lag s : - P : seasonal autoregressive (SAR) order - D : seasonal differencing order - Q : seasonal moving average (SMA) order - s : the seasonal period itself

Seasonal autoregressive (SAR) component. This captures how the current observation depends on observations from previous seasonal cycles. Its polynomial is:

$$\Phi_P(B^s) = 1 - \phi_1 B^s - \phi_2 B^{2s} - \dots - \phi_P B^{Ps}$$

So with $P = 1$ and monthly data, the model says "this January depends partly on last January."

Seasonal moving average (SMA) component. This captures how the current observation depends on forecast errors from previous seasonal cycles:

$$\Theta_Q(B^S) = 1 - \Theta_1 B^S - \Theta_2 B^{2S} - \dots - \Theta_Q B^{QS}$$

With $Q = 1$ and monthly data, the model says "this January's value is partly influenced by how far off the forecast was last January."

The full SARIMA model multiplies the non-seasonal and seasonal polynomials together, which is how it captures interactions between short-term dynamics and seasonal behavior.

Determining the Order of Seasonal Differencing

Choosing the right value of D is an important first step before fitting a SARIMA model. Here's how to approach it:

1.

Plot the series and look at the seasonal swings over time. - If the seasonal pattern stays roughly the same amplitude year after year, you likely need $D = 0$ (no seasonal differencing). - If the seasonal swings grow or shrink proportionally with the level of the series, that suggests $D = 1$.

2.

Check the ACF of the original series. If you see significant spikes at seasonal lags (lag 12, 24, 36 for monthly data) that decay very slowly, that's a strong signal you need seasonal differencing.

3.

Apply seasonal differencing and re-examine. After taking $D = 1$, plot the differenced series and its ACF again. The slow seasonal decay should be gone. If it persists, you might need $D = 2$, but this is uncommon. Over-differencing can actually make things worse by introducing artificial patterns, so be cautious.

A useful rule of thumb: if the standard deviation of the series decreases after seasonal differencing, the differencing was likely appropriate. If it increases, you may have over-differenced.

Interpreting Seasonal ARIMA Terms

Once you've fit a SARIMA model, interpreting the seasonal coefficients tells you about the nature of the seasonal patterns in your data.

SAR coefficients ($\Phi_1, \Phi_2, \dots, \Phi_P$) measure the strength and direction of the relationship between an observation and observations from previous seasonal cycles. A positive Φ_1 close to 1 means this year's value for a given season strongly resembles last year's value for the same season. A negative coefficient would mean the series tends to alternate between high and low values across seasonal cycles.

SMA coefficients ($\Theta_1, \Theta_2, \dots, \Theta_Q$) measure how forecast errors from past seasonal cycles influence the current value. These are often harder to interpret directly, but they play a critical role in capturing seasonal autocorrelation structure, especially when the ACF shows a sharp cutoff at seasonal lags rather than a slow decay.

Assessing significance. You evaluate whether each coefficient matters using t-tests or confidence intervals, just like with regular ARIMA terms. If a seasonal coefficient is not statistically significant, that term may not be contributing meaningfully to the model, and you should consider removing it for a more parsimonious fit. Comparing models with and without the term using AIC or BIC can also guide this decision.

9.2 Identifying seasonal patterns in ACF and PACF

Seasonal patterns in time series data repeat at fixed intervals, like yearly temperature changes. ACF and PACF plots help spot these patterns by showing significant spikes at lags that are multiples of the seasonal period.

To identify seasonal components, look for spikes at regular intervals in ACF and PACF plots. The lag of the first significant spike indicates the seasonal period. This helps distinguish between short-term fluctuations and long-term, recurring patterns in the data.

Identifying Seasonal Patterns in ACF and PACF

Seasonal patterns in ACF and PACF

- ACF and PACF plots help identify seasonal patterns in time series data which are regular, repeating patterns occurring at fixed intervals (temperature data showing yearly seasonality)
- ACF plot shows significant spikes at lags that are multiples of the seasonal period (lags 12, 24, 36 for monthly data with annual seasonality) with spikes gradually decreasing in magnitude as lag increases
- PACF plot shows significant spikes at lags that are multiples of the seasonal period with spikes cutting off abruptly after the seasonal lag (significant spike at lag 12 followed by insignificant spikes at lags 24, 36 for monthly data with annual seasonality)

Determining seasonal periods

- Seasonal period is the number of time steps between each repetition of the seasonal pattern (12 for monthly data with annual seasonality, 4 for quarterly data with annual seasonality)
- In ACF plots, the lag at which the first significant spike appears represents the seasonal period
- In PACF plots, the lag at which a significant spike appears followed by a cutoff in subsequent lags represents the seasonal period

Order of seasonal AR and MA terms

- Seasonal autoregressive (SAR) terms are identified using the PACF plot 1. The order of the SAR term is determined by the number of significant spikes at lags that are multiples of the seasonal period 2. A significant spike at lag 12 in the PACF plot indicates an SAR term of order 1, denoted as SAR(1)
- Seasonal moving average (SMA) terms are identified using the ACF plot 1. The order of the SMA term is determined by the number of significant spikes at lags that are multiples of the seasonal period 2. Significant spikes at lags 12 and 24 in the ACF plot indicate an SMA term of order 2, denoted as SMA(2)

Seasonal vs non-seasonal components

- Non-seasonal components are represented by significant spikes at short lags (lags 1, 2, 3) in both ACF and PACF plots indicating the presence of short-term dependencies in the time series data (daily stock price fluctuations)
- Seasonal components are represented by significant spikes at lags that are multiples of the seasonal period in both ACF and PACF plots indicating the presence of long-term, recurring patterns in the time series data (monthly retail sales data showing holiday seasonality)

- To distinguish between seasonal and non-seasonal components, analyze the ACF and PACF plots simultaneously
- Identify significant spikes at short lags (non-seasonal) and at lags that are multiples of the seasonal period (seasonal)
- Consider the magnitude and pattern of the spikes to determine the relative importance of seasonal and non-seasonal components (strong seasonal spikes with weak non-seasonal spikes suggest a predominantly seasonal time series)

9.3 Estimating and forecasting with SARIMA models

Model Estimation and Diagnostics

SARIMA models handle time series data that have both trend and seasonal patterns. They work by combining regular ARIMA components with seasonal counterparts, captured in the notation $(p, d, q) \times (P, D, Q)_m$, where m is the seasonal period (e.g., 12 for monthly data with yearly seasonality). This section covers how to estimate those parameters, check whether your model fits well, and generate forecasts.

Parameter Estimation for SARIMA Models

Maximum likelihood estimation (MLE) is the standard method for fitting SARIMA models. MLE finds the parameter values that make your observed data most probable under the model. The process works like this:

1. Specify the model order. Choose (p, d, q) for the non-seasonal part and $(P, D, Q)_m$ for the seasonal part, typically guided by ACF/PACF plots after differencing.
2. Initialize parameter values. The software sets starting guesses for each AR and MA coefficient (both regular and seasonal).
3. Optimize iteratively. An algorithm adjusts the parameters to maximize the likelihood function, repeating until the estimates stabilize (converge).
4. Check convergence. If the algorithm doesn't converge, you may need different starting values or a simpler model specification.

Other estimation methods exist, including conditional least squares (CLS) and unconditional least squares (ULS). In practice, most software defaults to MLE or a close variant, and for an intro course that's the one to know.

Diagnostic Tools for Model Fit

Once you've estimated a SARIMA model, you need to verify it actually captures the patterns in your data. Diagnostics focus on the residuals, which are the differences between observed values and what the model predicted.

What good residuals look like: - Uncorrelated with each other (no leftover pattern) - Approximately normally distributed - Mean of zero and roughly constant variance over time

Key diagnostic plots: - Residual ACF and PACF plots — If significant spikes remain, the model is missing some structure. This is often the most informative check. - Residuals vs. fitted values — Look for patterns or fanning (non-constant variance). - Q-Q plot — Compares residual distribution to a normal distribution. Points should fall near the diagonal line.

The Ljung-Box test provides a formal check for residual autocorrelation. Its null hypothesis is that the residuals are independently distributed (no autocorrelation). If you reject the null (low p-value), your model likely hasn't captured all the time series dynamics, and you should revisit your model specification.

Information criteria help you compare competing models: - AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion) both penalize model complexity while rewarding goodness-of-fit. Lower values are better. - BIC penalizes extra parameters more heavily than AIC, so it tends to favor simpler models. When AIC and BIC disagree, BIC's pick is more parsimonious.

Forecasting and Model Selection

Forecasting with SARIMA Models

Forecasting uses your estimated model to predict future values. SARIMA models produce two types of forecasts:

- Point forecasts give a single predicted value for each future time point.
- Prediction intervals give a range of plausible values around each point forecast. A 95% prediction interval means you expect the true value to fall within that range 95% of the time. These intervals widen as the forecast horizon increases, reflecting growing uncertainty further into the future.

The forecast horizon is how many periods ahead you're predicting. Short-horizon forecasts (1–3 steps ahead) are generally more accurate than long-horizon ones.

When interpreting SARIMA forecasts, keep in mind: - The forecasts will reflect the trend and seasonal patterns the model learned from historical data. - Prediction intervals communicate uncertainty. Narrow intervals suggest more confidence; wide intervals suggest less. - SARIMA assumes past patterns continue into the future. Structural breaks (e.g., a pandemic, a policy change) can invalidate forecasts quickly.

Model Selection and Comparison

Choosing the best SARIMA model means balancing fit and simplicity. You want a model that explains the data well without overfitting.

In-sample measures (how well the model fits the data it was trained on): - AIC and BIC (lower is better) - Residual diagnostics (clean residuals as described above)

Out-of-sample measures (how well the model predicts data it hasn't seen): - RMSE (Root Mean Squared Error) — penalizes large errors more heavily - MAE (Mean Absolute Error) — treats all errors equally by size - MAPE (Mean Absolute Percentage Error) — expresses error as a percentage, useful for comparing across different scales

Cross-validation techniques like rolling origin evaluation strengthen out-of-sample assessment. In rolling origin, you repeatedly fit the model on expanding windows of data and forecast the next observation, simulating real forecasting conditions.

A good model should perform well on both in-sample and out-of-sample metrics. If a model fits the training data perfectly but forecasts poorly, it's likely overfit.

Real-World Applications of SARIMA

SARIMA models are a natural fit for data with clear seasonal cycles. Common examples include:

- Retail sales — Monthly or quarterly sales data often show holiday spikes and seasonal buying patterns.

- Energy demand — Electricity consumption follows daily and seasonal cycles tied to weather and human activity.
- Economic indicators — Quarterly GDP growth or monthly inflation rates often exhibit seasonal regularities.

Steps for applying a SARIMA model in practice:

1. Plot the series and identify its characteristics (trend, seasonality, any obvious outliers).
2. Apply differencing (regular d and seasonal D) to achieve stationarity. Verify with ACF plots or unit root tests.
3. Use ACF and PACF plots of the differenced series to guide your choice of p, q, P, Q .
4. Estimate model parameters using MLE.
5. Run diagnostics: check residual plots, Ljung-Box test, and information criteria.
6. If diagnostics reveal problems, revise the model and re-estimate.
7. Generate forecasts and report prediction intervals alongside point forecasts.
8. Monitor performance over time and re-estimate as new data arrives.

Unit 10 – Multivariate Time Series Analysis

10.1 Vector Autoregression (VAR) models

Vector Autoregression (VAR) Models

Vector Autoregression (VAR) models extend the idea of autoregressive models from a single time series to multiple time series at once. Instead of modeling one variable in isolation, a VAR treats every variable in the system as depending on its own past values and the past values of all other variables. This makes VAR models especially useful for studying how interconnected variables like GDP, inflation, and unemployment evolve together over time.

Vector Autoregression Models in Time Series

A univariate AR model predicts one variable from its own lags. A VAR generalizes this: with k variables and P lags, each variable gets its own equation where it's regressed on P lagged values of every variable in the system.

For a two-variable VAR(1) with variables Y_t and X_t , the system looks like:

$$Y_t = c_1 + a_{11}Y_{t-1} + a_{12}X_{t-1} + e_{1t}$$

$$X_t = c_2 + a_{21}Y_{t-1} + a_{22}X_{t-1} + e_{2t}$$

Each equation has its own intercept, coefficients on all lagged variables, and an error term. The coefficients a_{12} and a_{21} are what capture the cross-variable dynamics that a univariate model would miss entirely.

Common applications of VAR models include:

- Forecasting multivariate time series where variables move together (GDP, inflation, unemployment)
- Analyzing feedback effects among variables (stock prices and exchange rates influencing each other)
- Studying the impact of shocks through impulse response functions (how a monetary policy shock ripples through the economy)
- Testing causal relationships using Granger causality tests (does money supply help predict inflation, or vice versa?)

Construction of VAR Models

Building a VAR model comes down to two main decisions: which variables to include, and how many lags to use.

Choosing the lag order P

The lag order P is the number of past time periods each equation looks back. Too few lags and you miss important dynamics; too many and you overfit the model and waste degrees of freedom (the number of parameters grows quickly with P and k).

To find the right P , you estimate the model at several candidate lag orders and compare them using:

- Akaike Information Criterion (AIC) and Schwarz Bayesian Information Criterion (SBIC/BIC): Both balance goodness of fit against model complexity. Lower values indicate a better trade-off. SBIC tends to favor more parsimonious (fewer-lag) models than AIC because it penalizes extra parameters more heavily.
- Likelihood ratio tests: Directly compare a model with p lags against one with $p - 1$ lags to see if the extra lag adds statistically significant explanatory power.

Estimating the model

Once you've chosen p , each equation in the VAR is estimated separately using ordinary least squares (OLS). Because every equation has the same set of right-hand-side variables (the same lags of all variables), estimating each equation by OLS individually is actually efficient. The estimated coefficients capture how each variable responds to past movements in itself and in the other variables.

Interpretation of VAR Coefficients

With multiple equations and potentially many lagged terms, interpreting raw VAR coefficients can get overwhelming. There are a few tools that help.

Coefficient interpretation

Each coefficient tells you the estimated effect of a one-unit increase in a lagged variable on the current value of the dependent variable, holding everything else constant. For example, in the two-variable system above, a_{12} tells you how a one-unit increase in last period's X is associated with a change in this period's Y . You can run standard t-tests on individual coefficients to check significance, or use joint F-tests to assess whether all lags of a particular variable matter for a given equation.

Impulse Response Functions (IRFs)

IRFs are often more informative than staring at individual coefficients. An IRF traces how a one-unit shock to one variable's error term affects all variables in the system over subsequent time periods.

For example, you could shock the oil price equation by one standard deviation and then track how GDP responds over the next 10 periods. The IRF might show GDP dipping for a few quarters before gradually returning to baseline. Confidence bands (often computed via bootstrapping) are plotted around the IRF to show how precisely the response is estimated. If the confidence band includes zero at a given horizon, the response isn't statistically significant at that point.

Diagnostic Tests for VAR Models

After estimation, you need to verify that the model is well-specified. Skipping diagnostics can lead to misleading forecasts and impulse responses.

Stability check

A VAR model is stable if the effects of shocks eventually die out rather than exploding over time. To check this, compute the eigenvalues of the companion matrix (a reformulation of the VAR system). All eigenvalues must have modulus (absolute value) strictly less than 1, meaning they lie inside the unit circle. If any eigenvalue sits on or outside the unit circle, the model is unstable, and its impulse responses and forecasts are unreliable.

Residual diagnostics

These tests check whether the model's assumptions hold:

- Lagrange Multiplier (LM) test: Tests for serial correlation (autocorrelation) in the residuals. Significant serial correlation suggests the model hasn't captured all the temporal dynamics.
- White test: Tests for heteroscedasticity, meaning the variance of the error terms changes over time. Constant error variance is an assumption of OLS.
- Jarque-Bera test: Tests whether the residuals follow a normal distribution by checking skewness and kurtosis.

Remedial actions if diagnostics fail

If any of these tests flag problems, you have several options:

1. Increase the lag order to capture dynamics the current model is missing
2. Add exogenous variables or deterministic terms (like a time trend or seasonal dummies) if there are systematic patterns the VAR isn't picking up
3. Transform the variables to achieve stationarity or stabilize variance (a logarithmic transformation is common for variables like GDP or prices that grow exponentially)

10.2 Granger causality

Granger causality is a powerful tool in multivariate time series analysis. It helps determine if one variable can predict another, offering insights into relationships between economic indicators, financial markets, and more. This concept goes beyond simple correlation, revealing potential causal links.

Using vector autoregressive models, Granger causality tests compare restricted and unrestricted models. The results can show unidirectional or bidirectional relationships, aiding in forecasting and policy analysis. However, it's crucial to remember that Granger causality doesn't imply true causality and has limitations.

Granger Causality in Multivariate Time Series Analysis

Concept of Granger causality

- Statistical concept determines if one time series helps forecast another (GDP and unemployment rate)
- Based on the idea that if variable X "Granger-causes" variable Y, then past values of X contain information that helps predict Y beyond the information in past values of Y alone (stock prices and company earnings)
- Relevant in multivariate time series analysis provides insights into causal relationships and dependencies among multiple variables over time
- Identifies variables with predictive power for other variables aids in understanding dynamics and interactions within a multivariate system (interest rates and inflation)
- Not necessarily true causality in the philosophical sense, but rather a statistical concept based on predictive ability
- Does not imply one variable directly causes another, but rather one variable contains information useful for predicting the other variable (weather patterns and crop yields)

Application of Granger causality tests

- Conducted using a vector autoregressive (VAR) model an extension of the univariate autoregressive model for multivariate time series

- In a VAR model, each variable is modeled as a linear function of its own past values and the past values of other variables in the system (stock prices, interest rates, and GDP)
- Involves estimating two VAR models: 1. The restricted model: Includes only the lagged values of the dependent variable (predicting GDP using only past GDP values) 2. The unrestricted model: Includes the lagged values of both the dependent variable and the potential causal variable (predicting GDP using past GDP and interest rate values)
- Compares the predictive power of the restricted and unrestricted models using an F-test or a likelihood ratio test
- If the unrestricted model provides a significantly better fit, it suggests the potential causal variable "Granger-causes" the dependent variable (interest rates Granger-cause GDP)
- Null hypothesis: the potential causal variable does not Granger-cause the dependent variable
- Rejecting the null hypothesis implies the potential causal variable has predictive power for the dependent variable (exchange rates Granger-cause trade balances)

Interpretation of Granger causality results

- Rejecting the null hypothesis indicates the potential causal variable has predictive power for the dependent variable
- Suggests past values of the potential causal variable contain information useful for forecasting the dependent variable, beyond the information in past values of the dependent variable alone (oil prices Granger-cause inflation)
- Direction of Granger causality can be unidirectional or bidirectional
- Unidirectional: variable X Granger-causes variable Y, but Y does not Granger-cause X (money supply Granger-causes inflation, but not vice versa)
- Bidirectional (feedback): variable X Granger-causes variable Y, and Y also Granger-causes X (stock prices and economic growth)
- Provides valuable insights into lead-lag relationships and information flow among variables in a multivariate system
- Used for forecasting, policy analysis, and understanding underlying system dynamics (monetary policy and economic indicators)

Limitations of Granger causality analysis

- Based on predictive ability, does not necessarily imply true causality in the philosophical sense
- Possible for a variable to Granger-cause another without a direct causal relationship (spurious correlation)
- Sensitive to the choice of lag length in the VAR model
- Different lag lengths can lead to different conclusions about Granger causality
- Important to use appropriate model selection criteria (AIC, BIC) to determine optimal lag length
- Assumes the underlying time series are stationary
- If non-stationary, results may be spurious

- Necessary to ensure stationarity (differencing or other transformations) before conducting tests
- Does not account for potential confounding variables or hidden factors influencing the relationship between variables
- Presence of confounding variables can lead to misleading conclusions (omitted variable bias)
- Based on linear models, may not capture non-linear causal relationships
- In the presence of strong non-linearities, tests may fail to detect important causal relationships (threshold effects or regime shifts)

10.3 Cointegration and error correction models

Cointegration and Error Correction Models

Cointegration helps you identify long-term equilibrium relationships between non-stationary time series. Two variables might each wander unpredictably on their own, but if they're cointegrated, they're tethered together over time. Error Correction Models (ECMs) then capture both the short-run dynamics and the mechanism that pulls variables back toward that long-run equilibrium.

Cointegration

Cointegration in Time Series

Two or more non-stationary time series are cointegrated if some linear combination of them produces a stationary series. In plain terms: each series individually has a unit root (trending mean, non-constant variance, etc.), but together they share a common stochastic trend and maintain a long-run equilibrium relationship.

- Deviations from this equilibrium are stationary and mean-reverting. Think of a stock price and its futures contract: they can diverge briefly, but the spread between them tends to snap back.
- Cointegrated series move together over the long run, preventing them from drifting arbitrarily far apart. Interest rates and inflation are a classic example.
- Because the equilibrium relationship is stable, you can estimate long-run parameters and measure how quickly variables adjust back to equilibrium after a shock.

The key distinction from ordinary correlation: correlation measures co-movement at a point in time, while cointegration describes a persistent structural relationship that holds across time, even when individual series are non-stationary.

Tests for Cointegrating Relationships

Engle-Granger Test — a two-step, residual-based approach:

1. Estimate the long-run equilibrium relationship using OLS: $y_t = \beta_0 + \beta_1 x_t + u_t$
2. Test the residuals u_t for stationarity using a unit root test (typically the Augmented Dickey-Fuller test). - If u_t is stationary, the series are cointegrated. For example, regressing household expenditure on income and finding stationary residuals would suggest cointegration.

This test is simple but limited to one cointegrating relationship between two variables.

Johansen Test — a maximum likelihood approach within a Vector Autoregressive (VAR) framework:

- Can test for multiple cointegrating relationships among several variables simultaneously (e.g., GDP, consumption, and investment).
- Works by examining the rank of the long-run coefficient matrix in the VAR. The rank tells you how many independent cointegrating vectors exist.
- Uses two test statistics: the trace statistic (tests whether the number of cointegrating vectors is at most r) and the maximum eigenvalue statistic (tests r vs. $r + 1$ cointegrating vectors).

Use Engle-Granger when you have two variables and want a quick check. Use Johansen when you have three or more variables or suspect multiple cointegrating relationships.

Error Correction Models (ECMs)

Error Correction Models

An ECM captures both short-run fluctuations and the long-run equilibrium in a single equation. The general form for two cointegrated variables is:

$$\Delta y_t = \alpha_0 + \alpha_1 \Delta x_t + \alpha_2 (y_{t-1} - \beta_0 - \beta_1 x_{t-1}) + \varepsilon_t$$

Breaking this apart:

- Δy_t and Δx_t are first differences, capturing short-run dynamics (e.g., period-to-period changes in stock prices).
- $(y_{t-1} - \beta_0 - \beta_1 x_{t-1})$ is the error correction term. It measures how far the system was from equilibrium in the previous period. If the stock-futures spread was unusually wide last period, this term picks that up.

Estimating an ECM (Engle-Granger two-step method):

1. Estimate the long-run relationship $y_t = \beta_0 + \beta_1 x_t + u_t$ using OLS. Save the residuals $\hat{u}_t$.
2. Estimate the ECM using OLS, plugging in the lagged residuals $\hat{u}_{t-1}$ as the error correction term:

$$\Delta y_t = \alpha_0 + \alpha_1 \Delta x_t + \alpha_2 \hat{u}_{t-1} + \varepsilon_t$$

Interpretation of ECM Parameters

- Short-run coefficient (α_1): The immediate impact of a one-unit change in X on Y . For instance, how a sudden change in income affects consumption this period.
- Long-run coefficient (β_1): The equilibrium relationship between the variables. If $\beta_1 = 0.8$ in a consumption-income model, then in the long run, a one-unit increase in income is associated with a 0.8-unit increase in consumption.
- Adjustment parameter (α_2): The speed at which Y corrects back toward equilibrium.
- This value should be negative and statistically significant for a valid ECM. A negative sign means the system corrects toward equilibrium rather than away from it.
- The larger the absolute value, the faster the adjustment. An α_2 of -0.5 means about half of any disequilibrium is corrected each period, while -0.1 means only 10% is corrected per period.

Applications of ECMs

Forecasting: ECMs tend to produce more accurate forecasts than models that ignore cointegration, particularly over longer horizons. The error correction term anchors predictions to the long-run equilibrium, preventing forecasts from drifting unrealistically. Exchange rate forecasting is a common application.

Policy analysis: ECMs let you separate the short-run and long-run effects of a policy change or shock. For example, you could model how a tax cut affects consumption immediately (short-run coefficient) versus after the economy fully adjusts (long-run coefficient). The adjustment parameter tells you how quickly the system returns to equilibrium after the intervention.

Impulse response functions derived from ECMs trace out the dynamic path of how a shock propagates through the system over time. You can visualize, for instance, how inflation responds period-by-period to a monetary policy shock, and how long it takes for the effect to fade as the system returns to equilibrium.

Unit 11 – Spectral Analysis in Time Series

11.1 Fourier analysis and periodogram

Fourier Analysis

Principles of Fourier analysis

The core idea behind Fourier analysis is that any time series can be broken down into a sum of sine and cosine waves, each with a different frequency. This transforms your data from the time domain (values over time) into the frequency domain (how much energy sits at each frequency).

Why does this matter? Many time series contain cyclical patterns that are hard to see just by looking at a plot. Fourier analysis pulls those cycles apart so you can identify them individually. Think of seasonal patterns in retail sales, daily temperature swings, or business cycles in economic data.

- Represents a time series as a combination of sinusoidal (trigonometric) components
- Detects hidden periodicities and cyclical behavior that may not be obvious in the raw data
- Widely applied in signal processing, physics, engineering, and economics

Discrete Fourier transform application

In practice, you're working with sampled data (discrete observations at regular intervals), so you use the Discrete Fourier Transform (DFT) rather than the continuous version. The DFT converts a finite sequence of N equally spaced samples into a set of complex-valued frequency components:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}$$

Here, x_n is the time series value at time step n , N is the total number of samples, and k is the frequency index. You can also go backwards: the inverse DFT reconstructs the original time series from its frequency components, so no information is lost.

Each frequency component X_k is a complex number that encodes two things:

- Amplitude ($|X_k|$): how strong that frequency is in the data. Larger amplitude means that frequency contributes more to the overall signal.
- Phase ($\arg(X_k)$): the timing offset of that frequency component. It tells you where in its cycle that wave starts relative to the beginning of your series.

Computing the DFT directly requires $O(N^2)$ operations, which gets slow for large datasets. The Fast Fourier Transform (FFT) algorithm reduces this to $O(N \log N)$, making Fourier analysis practical even for very long time series.

Periodogram

Periodogram for frequency identification

The periodogram estimates the power spectral density (PSD) of a time series. While the DFT gives you complex-valued frequency components, the periodogram boils those down to a single real number at each frequency: how much of the series' total variance lives at that frequency.

The formula is straightforward:

$$I(\omega_k) = \frac{1}{N} |X_k|^2$$

where $|X_k|^2$ is the squared magnitude of the DFT at frequency ω_k and N is the sample size.

When you plot the periodogram (power on the y-axis, frequency on the x-axis), you're looking for peaks. A tall peak at a particular frequency means the data has a strong cyclical component at that period. For example, a peak at frequency $1/12$ in monthly data points to an annual cycle.

- Peaks reveal dominant frequencies, including the fundamental frequency and its harmonics
- Useful for detecting diurnal cycles, seasonal patterns, or any hidden periodic behavior
- Serves as a key exploratory tool before fitting more formal spectral models

Limitations of periodogram analysis

The raw periodogram has several well-known problems you should be aware of:

Spectral leakage. If your time series doesn't contain an exact integer number of cycles for a given frequency, the power from that frequency "leaks" into neighboring frequency bins. This smears peaks and can make it harder to distinguish close frequencies. The standard fix is to apply a window function (such as Hamming or Hann) to the data before computing the DFT. These windows taper the endpoints of the series toward zero, reducing leakage at the cost of slightly wider peaks.

Inconsistent estimation. The periodogram is a statistically inconsistent estimator of the true PSD. Its variance does not shrink as you collect more data. Even with a very long series, the periodogram remains noisy. To get a smoother, more reliable estimate, you can:

- Average power across adjacent frequency bins (Daniell smoothing)
- Use data tapering or multitaper methods
- Apply Welch's method, which averages periodograms from overlapping segments

Limited frequency resolution. The finest frequency difference you can resolve is:

$$\Delta f = \frac{1}{T}$$

where T is the total duration of the time series. A longer observation window gives you finer resolution. You cannot improve resolution just by sampling faster; you need more total time.

11.2 Spectral density estimation

Spectral density estimation is a powerful tool for analyzing time series data. It breaks down complex signals into their frequency components, revealing hidden patterns and periodicities that might not be apparent in the raw data.

Choosing between parametric and non-parametric methods involves trade-offs in accuracy and flexibility. Smoothing techniques help balance the resolution-variance trade-off, allowing analysts to tailor their approach to the specific characteristics of their data.

Spectral Density Estimation

Concept of spectral density

- Spectral density, also known as power spectral density (PSD), describes the distribution of power or variance across different frequencies in a time series
- Provides information about the relative importance of different frequency components (low frequency, high frequency) in the series
- Represents the decomposition of the time series into a sum of sinusoidal components with different frequencies and amplitudes
- Fourier transform of the autocovariance function of a stationary time series
- Analyzing spectral density allows for identifying dominant frequencies or periodicities (seasonal patterns, business cycles), detecting hidden patterns or cycles, and understanding the relative contributions of different frequency components to the overall variability of the series

Parametric vs non-parametric estimation methods

- Parametric methods assume a specific model for the time series (autoregressive (AR), moving average (MA)) and estimate the parameters of the assumed model to calculate the spectral density
- Examples: Yule-Walker method, Burg method
- Advantages: provide smooth spectral density estimates, require fewer data points
- Disadvantages: rely on correct model specification, may not capture complex or non-linear relationships
- Non-parametric methods do not assume a specific model and estimate the spectral density directly from the data using techniques like periodogram or smoothed periodogram
- Examples: Bartlett method, Welch method
- Advantages: do not require assumptions about the underlying model, can capture complex or non-linear relationships
- Disadvantages: may produce noisy or less smooth estimates, require more data points

Smoothing techniques for spectral density

- Smoothing techniques reduce variance and improve stability of spectral density estimates
- Windowing multiplies the time series by a window function (Hamming, Hann, Bartlett) before computing the periodogram
- Reduces spectral leakage and improves resolution of the estimate
- Averaging divides the time series into overlapping or non-overlapping segments, computes the periodogram for each segment, and averages the periodograms
- Reduces variance at the cost of reduced frequency resolution
- Examples: Bartlett's method (non-overlapping segments), Welch's method (overlapping segments)

- Choice of window function and segment length depends on characteristics of the time series and desired trade-off between resolution and variance

Resolution vs variance trade-off

- Resolution: ability to distinguish between closely spaced frequency components in the spectral density estimate
- Higher resolution allows for better separation of nearby frequencies
- Variance: variability or noise in the spectral density estimate
- Lower variance leads to more stable and reliable estimates
- Inherent trade-off between resolution and variance in spectral density estimation
- Increasing resolution (longer window or more segments) typically increases variance
- Decreasing variance (shorter window or fewer segments) typically reduces resolution
- Choice of window length, segment length, and overlapping in smoothing techniques affects this trade-off
- Longer windows or segments improve resolution but increase variance
- Shorter windows or segments reduce variance but decrease resolution
- Optimal balance between resolution and variance depends on the specific application and characteristics of the time series being analyzed

11.3 Applications of spectral analysis

Spectral analysis is a powerful tool for understanding time series data across various fields. By decomposing complex signals into their frequency components, it reveals hidden patterns, cycles, and relationships that might not be apparent in the raw data.

From economics to engineering and geophysics, spectral analysis helps identify periodic components, trends, and noise in time series. It also enables the study of relationships between multiple time series, providing valuable insights for forecasting, control, and optimization in real-world applications.

Applications of Spectral Analysis

Application of spectral analysis techniques

- Apply spectral analysis techniques to real-world time series data from various fields
- Economics: Analyze business cycles (GDP fluctuations), stock market trends (S&P 500), and economic indicators (inflation rates)
- Engineering: Study vibrations (bridge oscillations), power systems (electrical grid frequency), and control systems (aircraft stability)
- Geophysics: Investigate seismic activity (earthquake waveforms), ocean waves (tidal patterns), and atmospheric phenomena (temperature variations)
- Steps to apply spectral analysis techniques: 1. Collect and preprocess the time series data by removing trends, outliers, and missing values 2. Compute the power spectral density (PSD) or periodogram using Fourier transform or Welch's method 3. Identify significant frequencies and their

corresponding powers from the PSD plot 4. Interpret the results in the context of the specific field and application to gain insights into underlying processes

Identification of time series components

- Spectral analysis decomposes a time series into its constituent frequencies and helps identify periodic components, trends, and noise
- Periodic components appear as peaks in the power spectrum at their corresponding frequencies (daily, weekly, or annual cycles)
- Trends manifest as low-frequency components in the power spectrum (long-term growth or decline)
- Noise is characterized by a continuous distribution of power across all frequencies (white noise or colored noise)
- Techniques for identifying components:
 - Periodogram: Estimate the power spectrum by computing the squared magnitude of the Fourier transform of the time series
 - Smoothing: Apply techniques like moving average or Hamming window to reduce noise and enhance significant peaks
 - Significance testing: Determine the statistical significance of peaks using methods like Fisher's g-test or the chi-squared test

Relationships between multiple time series

- Cross-spectral analysis extends spectral analysis to multiple time series and helps detect and quantify relationships between them
- Cross-periodogram: Compute the cross-power spectrum between two time series to identify common frequencies
- Coherence: Measure the linear relationship between two time series as a function of frequency
- Coherence ranges from 0 (no relationship) to 1 (perfect linear relationship) and helps assess the strength of the relationship
- Phase: Quantify the lead-lag relationship between two time series at each frequency to determine which series leads or lags the other
- Applications of cross-spectral analysis:
 - Identifying common periodic components or trends across multiple time series (stock prices and economic indicators)
 - Detecting and quantifying causal relationships or feedback loops between variables (temperature and CO2 levels)
 - Studying the propagation of signals or disturbances through a system (seismic waves through different layers of the Earth)

Interpretation of spectral analysis results

- Interpreting the power spectrum:

- Dominant frequencies indicate the presence of periodic components or cycles (annual sales patterns or tidal cycles)
- The power at each frequency represents the relative importance or strength of that component (amplitude of seasonal variations)
- Broad peaks suggest the presence of quasi-periodic behavior or noise (irregular business cycles or background seismic noise)
- Relating spectral results to the underlying processes:
 - Economic processes: Business cycles (3-5 years), seasonal patterns (quarterly or monthly), or long-term trends (decades)
 - Physical processes: Natural frequencies (bridge resonance), resonance (aircraft wing flutter), or energy transfer between components (coupled oscillators)
 - Geophysical processes: Tidal cycles (diurnal and semidiurnal), climate oscillations (El Niño), or seismic wave propagation (P-waves and S-waves)
- Combining spectral analysis with domain knowledge:
 - Use the insights gained from spectral analysis to refine models or theories (economic models incorporating cyclical components)
 - Identify potential causal relationships or feedback mechanisms (climate variables influencing each other)
 - Develop strategies for forecasting (predicting future sales based on seasonal patterns), control (damping vibrations in structures), or optimization (designing filters to remove noise) based on the spectral properties of the system

Unit 12 – State-Space Models & Kalman Filtering

12.1 Introduction to state-space models

State-space models are powerful tools for analyzing time series data. They consist of two main components: the state equation, which describes the evolution of unobserved variables, and the observation equation, which links these hidden states to observed measurements.

These models offer flexibility in handling complex patterns, estimating latent variables, and efficiently updating parameters. They're widely used in various applications, from univariate to multivariate time series, and can handle trends, seasonality, and structural breaks. Their versatility makes them invaluable in time series analysis.

State-Space Models

Components of state-space models

- State-space models powerful framework for analyzing and modeling time series data
- Consist of two main components: state equation and observation equation
- State equation describes evolution of unobserved state variables over time
- Captures dynamics of underlying process (stock market trends, population growth)
- Represents hidden states influencing observed time series
- Typically expressed as first-order difference equation or matrix equation
- Observation equation links unobserved state variables to observed time series measurements
- Defines relationship between hidden states and actual observations (stock prices, census data)
- Incorporates measurement errors or noise in observations
- Enables estimation of state variables based on observed data

Advantages of state-space models

- Flexibility in modeling complex time series patterns
- Handle wide range of behaviors (trends, seasonality, cycles)
- Allow incorporation of external factors or interventions affecting time series (policy changes, natural disasters)
- Ability to estimate unobserved components or latent variables
- Enable decomposition of time series into interpretable components (trend, seasonal, irregular)
- Provide framework for estimating and inferring hidden states or processes driving observed data
- Efficient parameter estimation and inference
- Estimated using powerful algorithms (Kalman filter, smoother)

- Enable recursive estimation and updating of model parameters as new data becomes available
- Facilitate computation of forecasts, smoothed estimates, and confidence intervals
- Handling of missing data and irregularly spaced observations
- Naturally accommodate missing observations or irregularly spaced data
- Kalman filter algorithm handles missing data by skipping update step for missing time points
- Integration with other modeling techniques
- Can be combined with other approaches (regression, time-varying parameter models)
- Provide unified framework for incorporating external covariates or explanatory variables

Applications in time series analysis

- Univariate time series
 - Applied to analyze and forecast single time series variable (stock prices, sales data, temperature measurements)
- Multivariate time series
 - Handle multiple time series variables simultaneously
 - Allow modeling of relationships and dependencies among different series (economic indicators, sensor data)
- Time series with trend and seasonality
 - Capture both trend and seasonal components
 - State variables represent underlying trend and seasonal factors
- Enable extraction and estimation of components for analysis and forecasting
- Time series with structural breaks or interventions
 - Incorporate structural breaks or interventions affecting dynamics (policy changes, economic shocks)
 - Include dummy variables or time-varying parameters to account for sudden changes or external events
- Time series with measurement errors or noise
 - Explicitly account for measurement errors or noise in observed data
 - Observation equation incorporates error term to capture discrepancy between true state and observed measurements

Formulation for specific problems

1. Identify the state variables - Determine unobserved components or hidden states governing dynamics of time series (trend, seasonality, cycles, other relevant variables)
2. Define the state equation - Specify transition matrix describing how state variables evolve over time - Determine appropriate form based on characteristics of time series - Include necessary parameters or coefficients

3. Define the observation equation - Specify relationship between state variables and observed time series measurements - Determine matrix mapping state variables to observations - Include measurement errors or noise terms
4. Specify the initial conditions - Determine initial values or distributions for state variables - Specify prior information or assumptions about initial state of system
5. Estimate the model parameters - Use techniques like maximum likelihood estimation or Bayesian inference - Employ algorithms (Kalman filter, smoother) to compute likelihood and update parameter estimates
6. Assess the model fit and performance
 - Evaluate goodness-of-fit using appropriate criteria (likelihood-based measures, residual diagnostics) - Compare performance with alternative models or benchmark methods - Validate forecasting accuracy using techniques (cross-validation, rolling-origin evaluation)

12.2 Kalman filter algorithm

Kalman Filter Algorithm

The Kalman filter estimates the hidden state of a dynamic system from noisy observations. It works recursively: at each time step, it predicts where the system should be, then corrects that prediction using new data. This predict-correct cycle is the core of the algorithm, and understanding it well will make state-space models click.

Purpose

The Kalman filter solves a fundamental problem: you can't directly observe the true state of a system, but you can observe noisy measurements related to that state. The filter gives you the best possible estimate (in a least-squares sense) by doing two things at every time step:

- Predicting the next state based on a model of how the system evolves
- Updating that prediction when a new observation arrives, weighting the correction by how much you trust the observation versus the model

Uncertainty is tracked throughout using error covariance matrices, so you always know how confident the estimate is. Applications range from GPS tracking to weather forecasting to financial modeling.

The State-Space Setup

Before running the filter, you need two equations that define your system.

State transition equation (how the hidden state evolves):

$$x_t = Fx_{t-1} + w_t, \text{ where } w_t \sim N(0, Q)$$

- F is the state transition matrix, encoding how the previous state maps to the current one
- w_t is process noise, capturing model imperfections, with covariance Q

Observation equation (how measurements relate to the state):

$$z_t = Hx_t + v_t, \text{ where } v_t \sim N(0, R)$$

- H is the observation matrix, mapping the state into measurement space

- V_t is observation noise, with covariance R

Together, F , Q , H , and R fully specify the linear Gaussian model the Kalman filter operates on.

Prediction and Update Steps

This is the heart of the algorithm. Each time step has two phases.

Prediction step (project the state and covariance forward):

1. Predicted state estimate: $\hat{X}_{t|t-1} = F\hat{X}_{t-1|t-1}$
2. Predicted error covariance: $P_{t|t-1} = FP_{t-1|t-1}F^T + Q$

The predicted covariance grows here because process noise (Q) gets added. The model alone always makes you less certain.

Update step (correct the prediction using the new observation Z_t):

1. Kalman gain: $K_t = P_{t|t-1}H^T(HP_{t|t-1}H^T + R)^{-1}$
2. Updated state estimate: $\hat{X}_{t|t} = \hat{X}_{t|t-1} + K_t(z_t - H\hat{X}_{t|t-1})$
3. Updated error covariance: $P_{t|t} = (I - K_tH)P_{t|t-1}$

The term $(z_t - H\hat{X}_{t|t-1})$ is called the innovation (or measurement residual). It's the gap between what you observed and what you predicted you'd observe. The Kalman gain K_t controls how aggressively you correct toward the observation.

How to think about the Kalman gain: When observation noise R is small relative to the predicted uncertainty $P_{t|t-1}$, the gain is large and the filter trusts the measurement more. When R is large (noisy sensors), the gain shrinks and the filter leans on the model prediction instead. The filter automatically balances these two sources of information.

Prediction vs. Updating vs. Smoothing

These three operations differ in which observations they use:

- Prediction (filtering forward): Estimates the state at time t using only observations up to $t - 1$. This is what you do before the new measurement arrives.
- Updating (filtering): Estimates the state at time t using observations up to and including t . This is the corrected estimate after incorporating the latest data.
- Smoothing: Estimates the state at time t using all observations, including those from times after t . Since it uses future information, smoothing produces more accurate estimates but can only be done retrospectively.

Two common smoothing variants: - Fixed-interval smoothing re-estimates all states over a complete time window after all data is collected - Fixed-lag smoothing estimates the state a fixed number of steps in the past, useful when you can tolerate a small delay

Implementation Steps

To implement the Kalman filter in practice (using Python with NumPy, MATLAB, R, etc.):

1. Define the model matrices F , Q , H , and R based on your system

2. Set initial conditions: choose an initial state estimate $\hat{x}_{0|0}$ and initial error covariance $P_{0|0}$. If you're unsure about the initial state, set $P_{0|0}$ large to reflect that uncertainty
3. Loop through each time step: - Run the prediction step to get $\hat{x}_{t|t-1}$ and $P_{t|t-1}$ - Compute the Kalman gain K_t - Run the update step to get $\hat{x}_{t|t}$ and $P_{t|t}$
4. Store results (state estimates and covariances) at each step for later analysis or plotting
5. Validate by testing on simulated data where you know the true state, then compare your filtered estimates against ground truth

A common sanity check: the innovation sequence ($z_t - H\hat{x}_{t|t-1}$) should look like white noise if your model is well-specified. If it shows patterns, your model matrices likely need adjustment.

12.3 Applications of state-space models and Kalman filtering

State-space models and Kalman filtering provide a framework for estimating hidden (unobserved) variables in systems that change over time. They combine a model of how the system evolves with noisy observations to produce optimal estimates of the true underlying state. This makes them valuable across finance, engineering, and the natural sciences, especially when data is multivariate, noisy, or contains gaps.

Compared to traditional approaches like ARIMA or exponential smoothing, state-space models handle more complex structures naturally: multiple related variables, time-varying parameters, and missing observations are all built into the framework rather than bolted on.

Applications and Evaluation of State-Space Models and Kalman Filtering

Applications of state-space models

Finance. State-space models are widely used in finance because many quantities of interest (true volatility, risk exposures, fair value) aren't directly observable. You observe market prices and returns, then use the model to infer what's happening underneath.

- Portfolio optimization — estimating how optimal asset allocations shift over time as market conditions change
- Asset pricing — estimating the fair value of stocks or bonds based on latent economic factors that drive returns
- Risk management — tracking evolving financial risks (market risk, credit risk) to inform hedging and capital decisions
- Volatility forecasting — predicting future price fluctuations for options pricing and trading strategies, where volatility itself is an unobserved state

Engineering. In engineering, the "state" is often a physical quantity like position, velocity, or orientation that you can only measure indirectly through noisy sensors.

- Control systems — representing and optimizing dynamic system behavior in robotics and aerospace
- Navigation and guidance — estimating position, velocity, and orientation of aircraft or spacecraft from sensor readings (GPS, accelerometers, gyroscopes) and fusing those measurements optimally

- Robotics — modeling how robots move and interact with their environment, enabling autonomous vehicles and industrial automation
- Signal processing — filtering and estimating signals buried in noise, with applications in speech recognition and image processing

Natural sciences. Many natural systems have hidden states (true atmospheric conditions, population sizes, disease prevalence) that are only partially observed.

- Weather forecasting — predicting atmospheric conditions by combining physical models of the atmosphere with sparse, noisy observations from weather stations and satellites
- Oceanography — modeling ocean currents, waves, and their interactions with climate systems
- Ecology — tracking population dynamics and species interactions (e.g., predator-prey relationships) where direct counts are unreliable
- Epidemiology — estimating the true spread of infectious diseases like COVID-19 or influenza from incomplete case reports, and evaluating intervention effectiveness

Forecasting with Kalman filtering

The Kalman filter operates on two equations that together define the state-space model:

- State equation: $x_t = F_t x_{t-1} + v_t$ — describes how the hidden state evolves over time. F_t is the state transition matrix, and v_t is process noise (random shocks to the system).
- Observation equation: $y_t = H_t x_t + w_t$ — links what you actually observe (y_t) to the hidden state. H_t is the observation matrix, and w_t is measurement noise.

The filter itself alternates between two steps at each time point:

Step 1: Prediction. Use the model to project forward from the last known state.

1. Predict the state: $\hat{x}_{t|t-1} = F_t \hat{x}_{t-1|t-1}$ This takes your best estimate from the previous time step and applies the transition model.
2. Predict the covariance: $P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q_t$ This tracks how uncertain you are about the prediction. Q_t is the process noise covariance, which adds uncertainty because the system is subject to random shocks.

Step 2: Update. Incorporate the new observation to correct the prediction.

1. Compute the Kalman gain: $K_t = P_{t|t-1} H_t^T (H_t P_{t|t-1} H_t^T + R_t)^{-1}$ This determines how much weight to give the new observation versus the prediction. When measurement noise (R_t) is large relative to prediction uncertainty, K_t is small and you trust the prediction more. When R_t is small, you trust the observation more.
2. Update the state: $\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t (y_t - H_t \hat{x}_{t|t-1})$ The term $(y_t - H_t \hat{x}_{t|t-1})$ is the innovation — the difference between what you observed and what you predicted. The Kalman gain scales this correction.
3. Update the covariance: $P_{t|t} = (I - K_t H_t) P_{t|t-1}$ Incorporating the observation always reduces uncertainty, so $P_{t|t}$ is smaller than $P_{t|t-1}$.

Forecasting and missing values:

- To forecast beyond the current time, recursively apply the prediction step without any update (since there are no future observations yet).
- To handle missing values, use the Kalman smoother, which runs a backward pass through the data after the forward filter. It uses future observations to refine past state estimates, effectively interpolating through gaps.

Performance evaluation of models

Accuracy metrics quantify how close your predictions are to actual values:

- MSE (Mean Squared Error) — average of squared prediction errors. Penalizes large errors heavily.
- RMSE (Root Mean Squared Error) — square root of MSE, so it's in the same units as your data. Easier to interpret than MSE.
- MAE (Mean Absolute Error) — average of absolute prediction errors. Less sensitive to outliers than MSE because it doesn't square the errors.
- MAPE (Mean Absolute Percentage Error) — expresses errors as percentages of actual values. Useful for comparing accuracy across different scales, but can be misleading when actual values are near zero.

Diagnostic tools check whether the model's assumptions hold:

- Residual analysis — the prediction errors (residuals) should be approximately normal, have constant variance (homoscedasticity), and show no systematic patterns over time.
- ACF and PACF of residuals — if the autocorrelation or partial autocorrelation functions show significant spikes, there's temporal structure the model hasn't captured.
- Q-Q plots and histograms — visual checks for whether residuals follow a normal distribution. Deviations in the tails of a Q-Q plot suggest the model underestimates extreme events.
- Ljung-Box test — a formal statistical test for autocorrelation in the residuals. A significant result (small p-value) means the residuals still contain predictable structure, indicating the model is inadequate.

State-space models vs other techniques

Feature	ARIMA	Exponential Smoothing	Machine Learning	State-Space / Kalman
Multivariate data	Limited	No	Yes	Yes
Missing values	Difficult	Difficult	Requires imputation	Handled naturally
Time-varying parameters	No	Limited	Possible	Yes
Exogenous variables	ARIMAX variant	Limited	Yes	Yes
Interpretability	Moderate	High	Low (often black-box)	High
Data requirements	Moderate	Low	High	Moderate

ARIMA works well for univariate series but requires stationarity (often achieved through differencing, which can lose information). It struggles with multiple related series and doesn't naturally accommodate missing data.

Exponential smoothing is simple to implement and effective for short-term forecasts where trend and seasonality are stable. It's limited when patterns are complex or time-varying.

Machine learning (neural networks, SVMs) can capture complex nonlinear relationships but requires large training datasets, is prone to overfitting without careful regularization, and produces models that are difficult to interpret in terms of underlying processes.

State-space models offer a middle ground: they're flexible enough to handle multivariate data, missing values, and time-varying dynamics while remaining interpretable. The main trade-off is that you need to specify the model structure (the F_t , H_t , Q_t , and R_t matrices), which requires domain knowledge about the system you're modeling.

Unit 13 – Time Series Regression & Intervention Analysis

13.1 Regression with time series data

Regression with Time Series Data

Standard regression assumes that each observation is independent of the others. Time series data breaks that assumption because observations are ordered in time, and nearby values tend to be correlated. This creates problems you won't encounter with cross-sectional data, so time series regression requires its own set of tools and checks.

Challenges in Time Series Regression

Autocorrelation is the core issue. When today's value is correlated with yesterday's value (and the day before that), your observations aren't independent. If you ignore autocorrelation and run ordinary regression anyway, your coefficient estimates may still be unbiased, but the standard errors will be wrong. That means your confidence intervals and hypothesis tests can't be trusted.

Non-stationarity is the other major challenge. A stationary series has a constant mean, variance, and autocovariance over time. Many real-world series aren't stationary: stock prices trend upward, temperatures follow seasonal cycles, and so on. Regressing one non-stationary series on another can produce spurious regression, where you find a strong statistical relationship between variables that have no real connection. The classic example is two unrelated series that both happen to trend upward over time.

Seasonality and trends also need explicit treatment. If your data has a recurring monthly pattern or a long-run upward drift and you don't account for it, those patterns get absorbed into your error term. The result is biased coefficients and poor predictions.

Components of Time Series Models

A time series regression model typically accounts for three types of structure:

- **Trend component:** Captures the long-term direction of the series. A simple approach is to include a time index t as a predictor (linear trend). For curves, you might use t^2 (quadratic) or a log transformation. The choice depends on what the data actually looks like when you plot it.
- **Seasonality component:** Represents repeating patterns at fixed intervals (monthly, quarterly, etc.). You can model this with dummy variables (one for each season minus one) or with Fourier terms (pairs of sine and cosine functions). Fourier terms are especially useful when the seasonal pattern is smooth rather than sharply different across periods.
- **Exogenous variables:** External factors that influence your series but aren't part of its own past behavior. Examples include GDP growth affecting retail sales, a policy change shifting energy consumption, or a marketing campaign boosting website traffic. These can be time-varying or constant (like a binary indicator for "before vs. after" a one-time event).

Techniques for Non-Stationarity and Autocorrelation

Differencing removes non-stationarity in the mean. First-order differencing creates a new series:

$$\Delta y_t = y_t - y_{t-1}$$

Each value becomes the change from the previous period rather than the level. If the series still isn't stationary after first differencing, you can difference again (second-order differencing). For seasonal non-stationarity, you'd use a seasonal difference: $y_t - y_{t-s}$, where s is the seasonal period (e.g., 12 for monthly data).

Detrending is an alternative to differencing when the non-stationarity comes from a deterministic trend. You estimate the trend (say, by regressing y_t on t), then subtract it from the original series. The residuals from that regression become your detrended series, which you can then model as stationary.

Differencing vs. detrending: Use differencing when the trend is stochastic (random-walk-like behavior). Use detrending when the trend is deterministic (a predictable function of time). Choosing the wrong one can distort your results.

Lagged dependent variables address autocorrelation directly. By including past values of y as predictors (e.g., y_{t-1} , y_{t-2}), you let the model capture the temporal dependence. These are called autoregressive terms. The number of lags to include is a modeling decision, often guided by the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots.

Model Evaluation and Prediction

Evaluation of Time Series Models

Residual analysis is your first check after fitting a model. For the model to be adequate, the residuals should behave like white noise:

- No autocorrelation (check with the ACF plot of residuals)
- Constant variance over time (homoscedasticity)
- Approximately normal distribution

The Durbin-Watson test specifically tests for first-order autocorrelation in residuals. The test statistic ranges from 0 to 4, with values near 2 indicating no autocorrelation. Values well below 2 suggest positive autocorrelation; values well above 2 suggest negative autocorrelation. If your residuals are autocorrelated, the model is missing some structure in the data.

Information criteria help you choose between competing models. Both AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion) balance goodness-of-fit against model complexity. Lower values are better. BIC penalizes extra parameters more heavily than AIC, so it tends to favor simpler models.

Out-of-sample forecasting tests whether your model actually predicts well on data it hasn't seen:

1. Split your data into a training set (earlier observations) and a test set (later observations). Don't shuffle randomly; the time order matters.
2. Fit the model on the training set only.
3. Generate forecasts for the test set period.
4. Compare forecasts to actual values using accuracy metrics like RMSE (root mean squared error), MAE (mean absolute error), or MAPE (mean absolute percentage error).

Rolling window cross-validation takes this further. Instead of a single train/test split, you repeatedly move the training window forward in time, forecast the next period, and record the error. This gives you a more robust picture of how the model performs across different time periods, not just one arbitrary split point.

13.2 Autocorrelated errors and generalized least squares

Time series regression often faces autocorrelated errors, where error terms are linked across time periods. This can lead to issues with OLS estimators, including underestimated standard errors and inflated R-squared values, affecting hypothesis tests and forecasts.

Generalized least squares (GLS) offers a solution by transforming the model to account for autocorrelation. This method estimates parameters more efficiently, yielding unbiased estimates and correct standard errors, which allows for valid inference and improved model fit.

Autocorrelated Errors in Time Series Regression

Autocorrelation in time series regression

- Error terms correlated with each other across time in a time series regression model
- Positive autocorrelation: Errors tend to have the same sign as errors in the previous period (stock prices)
- Negative autocorrelation: Errors tend to have the opposite sign as errors in the previous period (temperature fluctuations)
- Consequences of autocorrelated errors lead to issues with OLS estimators
- No longer the best linear unbiased estimators BLUE
- Underestimated standard errors of coefficients result in invalid hypothesis tests and confidence intervals (t-tests, F-tests)
- Inflated R-squared values give a false sense of model fit (spurious regression)
- Inaccurate predictions and forecasts based on the model (weather forecasting, stock market predictions)

Generalized least squares for autocorrelation

- Method to estimate parameters of a linear regression model with autocorrelated errors
- Transforms model to account for autocorrelation structure in errors
- Transformed model satisfies assumptions of classical linear regression model (homoscedasticity, no autocorrelation)
- Applying GLS involves multiple steps 1. Estimate autocorrelation structure of errors using residuals from OLS regression (Durbin-Watson test) 2. Transform original data using estimated autocorrelation structure (Cochrane-Orcutt procedure) 3. Estimate regression model using transformed data
- Yields efficient and unbiased parameter estimates in presence of autocorrelated errors (improved statistical properties)

Estimation and Interpretation of GLS Models

Estimation of GLS models

- Obtain transformed data based on estimated autocorrelation structure
- Apply OLS to transformed data to estimate model parameters

- Interpret GLS model parameters
- Coefficients represent change in dependent variable for one-unit change in independent variable, holding other variables constant
- Interpretation similar to OLS, but coefficients adjusted for autocorrelation in errors (weather patterns, stock returns)
- Hypothesis tests and confidence intervals for GLS model parameters
- Use standard errors of coefficients from GLS estimation
- Interpret results the same as OLS, but estimates adjusted for autocorrelation (t-tests, p-values)

OLS vs GLS for autocorrelated errors

- OLS performance with autocorrelated errors
- Estimates unbiased but inefficient
- Underestimated standard errors lead to invalid inference (t-tests, confidence intervals)
- Inflated R-squared values
- GLS performance with autocorrelated errors
- Estimates unbiased and efficient
- Correctly estimated standard errors allow for valid inference
- R-squared values not inflated, provide more accurate measure of model fit
- Comparing OLS and GLS
- GLS outperforms OLS with autocorrelated errors, providing more accurate and efficient estimates (economic models, climate data)
- OLS used when errors not autocorrelated, simpler to implement and interpret (cross-sectional data)

13.3 Intervention analysis and modeling structural breaks

Intervention Analysis and Structural Breaks

Intervention analysis is a method for measuring how specific external events (like new laws, natural disasters, or product launches) affect a time series. Without it, your model treats these disruptions as unexplained noise, which leads to biased parameter estimates and poor forecasts.

Concept of Intervention Analysis

A time series sometimes gets hit by an outside event that changes its behavior. Intervention analysis gives you a formal way to detect, quantify, and account for that change.

The core goals are:

- Determine significance: Did the intervention actually affect the series, or is the apparent shift just random variation?
- Quantify the effect: How large was the impact, and in which direction?

- Characterize duration: Was the effect temporary (a one-week spike) or permanent (a new baseline level)?
- Improve forecasts: Models that ignore real interventions will systematically over- or underestimate future values.

Interventions are always external to the series itself. A new tax law, an earthquake, or a competitor entering the market are interventions. A gradual trend that was already present in the data is not.

Modeling Structural Breaks

A structural break is an abrupt change in the level, trend, or variance of a time series. You model these breaks by adding dummy variables to your regression or ARIMA specification.

There are two fundamental types of intervention functions:

Pulse function (temporary effect)

$$P_t = \begin{cases} 1 & \text{if } t = t_0 \\ 0 & \text{otherwise} \end{cases}$$

Use this when the event has a one-time impact that doesn't persist. For example, a factory shutting down for a single week of maintenance would cause a temporary drop in output at t_0 , with production returning to normal afterward.

Step function (permanent effect)

$$S_t = \begin{cases} 1 & \text{if } t \geq t_0 \\ 0 & \text{otherwise} \end{cases}$$

Use this when the event shifts the series to a new level permanently. A new emissions regulation that caps factory output going forward is a good example.

Once you've chosen the right function, you incorporate it directly into your model. Here's an ARIMA(1,0,0) example with a pulse intervention:

$$y_t = \beta_0 + \beta_1 P_t + \phi_1 y_{t-1} + \varepsilon_t$$

- β_0 is the intercept (baseline level)
- β_1 captures the size of the intervention effect
- ϕ_1 is the autoregressive coefficient
- ε_t is white noise error

If you suspected a permanent break instead, you'd replace P_t with S_t . You can also include both types simultaneously if the series experienced multiple interventions.

Assessing the Impact of Interventions

After estimating the model, you need to evaluate whether the intervention coefficients are meaningful.

Steps for assessment:

1. Estimate parameters using your chosen method (e.g., maximum likelihood for ARIMA models).
2. Test significance of each intervention coefficient with a t-test. A p-value below 0.05 suggests the intervention had a real effect on the series.

3. Interpret the coefficient's sign and size. A positive β_1 means the intervention raised the series level; a negative β_1 means it lowered it. The magnitude tells you by how much. - Example: A new marketing campaign with $\beta_1 = 1000$ means the campaign is associated with an increase of 1,000 units in weekly sales. - Example: A competitor's product launch with $\beta_1 = -0.05$ (in a proportion-scaled model) means a 5 percentage point drop in market share.
4. Assess duration. If you used a pulse function and it's significant, the effect was temporary. If a step function is significant, the series has shifted to a new permanent level.

Applying Intervention Analysis in Practice

Putting this together on real data involves a few key decisions:

1. Identify candidate interventions. Use domain knowledge (you know a regulation took effect on a certain date) or visual inspection of the time series plot (a sudden jump or drop that doesn't fit the prior pattern).
2. Choose the right dummy variable. One-time events like a store renovation get a pulse function. Lasting changes like a new tax policy get a step function.
3. Estimate the model and check significance. If β_1 is significant, the intervention matters. If not, the apparent break may just be noise.
4. Use results for forecasting. Once you've quantified a past intervention (e.g., a regulation reduced factory output by 10%), you can adjust future projections accordingly. For instance, if a similar regulation is expected, you can incorporate that estimated effect into your forecast.

Limitations to keep in mind:

- Exogeneity assumption. The intervention must be independent of past values of the series. If the "intervention" was actually triggered by the series itself (e.g., a policy enacted because prices spiked), the analysis breaks down.
- Confounding events. If two interventions happen at the same time (say, a holiday season and a new product launch overlap), separating their individual effects becomes difficult. You'd need additional information or assumptions to disentangle them.
- Known timing. Intervention analysis assumes you know when the break occurred. If the break point is unknown, you'll need a structural break test (like the Chow test) to detect it first.

Unit 14 – Volatility Models: ARCH and GARCH

14.1 Characteristics of volatility in time series

Understanding Volatility in Time Series

Definition and Importance of Volatility

Volatility measures how much a time series fluctuates over time. More precisely, it quantifies the degree of variation in a series, capturing the uncertainty or risk tied to how large those changes can be. Think of stock prices or exchange rates: some periods are calm, others are wild. Volatility puts a number on that difference.

Why does this matter? Volatility is central to:

- Risk management — understanding how much a portfolio's value could swing
- Portfolio optimization — balancing expected returns against potential losses
- Pricing financial derivatives — options and futures prices depend directly on expected volatility

Beyond finance, volatility shows up in economics, environmental science, and anywhere that uncertainty in a time series affects decisions.

Key Characteristics of Financial Volatility

Two patterns show up again and again in real financial data, and both are reasons why simple models fall short.

Volatility clustering means that large changes tend to follow large changes, and small changes tend to follow small changes. If you look at daily stock returns, you'll notice stretches of high turbulence bunched together, then stretches of relative calm. This isn't random — it implies that volatility is persistent and autocorrelated. A volatile day makes tomorrow more likely to be volatile too.

Asymmetry (the leverage effect) means that negative shocks move volatility more than positive shocks of the same size. A 3% drop in a stock index typically increases future volatility more than a 3% gain decreases it. This is why market crashes produce outsized spikes in volatility compared to equivalent rallies. The term "leverage effect" comes from the idea that falling prices raise a firm's debt-to-equity ratio, making it riskier.

Modeling Volatility in Time Series

Concept of Heteroskedasticity

Traditional time series models like ARMA and ARIMA assume homoskedasticity, meaning the variance of the error terms stays constant over time: $\sigma_t^2 = \sigma^2$ for all t .

Heteroskedasticity is when that assumption breaks down — the variance of the error terms changes over time: $\sigma_t^2 \neq \sigma^2$. In financial data, this is the norm, not the exception. Calm periods have small error variance; turbulent periods have large error variance.

Why you can't just ignore it:

- Parameter estimates become inefficient (not as precise as they could be)
- Confidence intervals and hypothesis tests become unreliable
- Forecasts miss the changing risk level entirely

This is exactly why specialized models like ARCH and GARCH were developed — they let the variance itself be a function of past data, rather than treating it as fixed.

Limitations of Traditional Volatility Models

ARMA and ARIMA models are powerful tools for modeling the level of a time series, but they have real blind spots when it comes to volatility:

- Constant variance assumption — They treat variance as fixed, so they can't represent periods of high vs. low volatility.
- No volatility clustering — They have no mechanism to capture the persistence where turbulent periods stick together and calm periods stick together.
- Symmetric treatment of shocks — A positive surprise and a negative surprise of the same size produce identical effects, so the leverage effect goes completely unmodeled.

These gaps motivated the development of ARCH and GARCH models, which explicitly model time-varying variance. ARCH lets today's variance depend on past squared errors, and GARCH extends that by also including past variance values. Together, they capture clustering and persistence. Extensions like EGARCH and GJR-GARCH go further by incorporating asymmetry.

14.2 ARCH models and their properties

ARCH Models

ARCH models explain why financial time series like stock returns and exchange rates go through alternating stretches of calm and turbulence. By letting the variance of the error term change over time, these models formalize volatility clustering, one of the most robust stylized facts in finance. This section covers the model structure, how parameters are estimated, how to read the output, and where ARCH falls short.

Structure of ARCH models

A standard time series regression assumes constant variance in the error term. ARCH relaxes that assumption by making the variance depend on recent squared errors.

An ARCH(q) model has two equations:

Conditional mean equation:

$$y_t = \mu_t + \varepsilon_t$$

- μ_t is the conditional mean of y_t given past information. It could be a constant, a moving average, or a regression model.
- ε_t is the error term, whose variance is not constant but instead changes over time.

Conditional variance equation:

$$\sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \alpha_2 \varepsilon_{t-2}^2 + \dots + \alpha_q \varepsilon_{t-q}^2$$

- $\alpha_0 > 0$ and $\alpha_i \geq 0$ for $i = 1, \dots, q$. These constraints guarantee that the conditional variance stays positive.
- q is the order of the model. An ARCH(1) uses only the most recent squared error; an ARCH(3) looks back three periods.

The error term is assumed conditionally normal:

$$\varepsilon_t \mid \Omega_{t-1} \sim N(0, \sigma_t^2)$$

where Ω_{t-1} is the information set available at time $t-1$ (past returns, past volatility, etc.). Notice that unconditionally the distribution of ε_t will have heavier tails than a normal, which aligns with what we actually see in financial returns.

The key intuition: when a large shock (positive or negative) hits at time $t-1$, the squared error ε_{t-1}^2 is large, which pushes σ_t^2 up. That's volatility clustering in equation form.

Parameter estimation for ARCH

ARCH parameters are estimated by maximum likelihood estimation (MLE).

The log-likelihood function for an ARCH(q) model is:

$$\ln L(\theta) = -\frac{1}{2} \sum_{t=1}^T \left[\ln(2\pi) + \ln(\sigma_t^2) + \frac{\varepsilon_t^2}{\sigma_t^2} \right]$$

- $\theta = (\mu, \alpha_0, \alpha_1, \dots, \alpha_q)$ is the full parameter vector.
- T is the number of observations (daily, weekly, or monthly returns).

The estimation process in practice:

1. Choose starting values for θ .
2. For each observation t , compute $\varepsilon_t = y_t - \mu_t$ and then σ_t^2 from the variance equation.
3. Evaluate the log-likelihood using the formula above.
4. Use a numerical optimizer (common choices: BFGS, Nelder-Mead) to find the θ that maximizes $\ln L$.
5. Obtain standard errors from the inverse of the Hessian matrix (the matrix of second derivatives of the log-likelihood) evaluated at the optimum.

Because there's no closed-form solution, software does the heavy lifting. But understanding the steps helps you diagnose convergence problems when they arise.

Interpretation of ARCH coefficients

- α_0 represents the baseline level of variance. It's the portion of volatility that persists regardless of recent shocks.
- α_i measures how much the squared error from i periods ago contributes to today's conditional variance. A larger α_i means past shocks of that lag have a stronger influence on current volatility.

For an ARCH(1), the unconditional variance of ε_t is $\frac{\alpha_0}{1-\alpha_1}$, which only exists when $\alpha_1 < 1$. If α_1 is close to 1, shocks die out very slowly and volatility is highly persistent.

Goodness-of-fit tools:

- Likelihood ratio test compares the ARCH model against a restricted constant-variance model. A significant test statistic means the ARCH terms add explanatory power.
- AIC and BIC balance fit against complexity. Lower values indicate a better model. BIC penalizes extra parameters more heavily than AIC, so it tends to favor more parsimonious specifications.
- Ljung-Box test on squared standardized residuals checks whether the model has captured all the volatility clustering. If the test is significant, autocorrelation remains in the squared residuals, and you may need a higher order or a different model.

Limitations of ARCH models

ARCH models are a foundational tool, but they have well-known drawbacks:

- Non-negativity constraints are restrictive. Every α_i must be non-negative. In practice this can prevent the model from fitting certain volatility patterns, especially when the number of lags is large.
- High orders are often needed. Capturing slow-decaying volatility persistence may require a large q , which leads to many parameters and potential convergence difficulties during estimation. This is actually the main motivation for GARCH models, which achieve long memory with far fewer parameters.
- Symmetric treatment of shocks. The variance equation uses ε_{t-1}^2 , so a positive shock and a negative shock of the same size have identical effects on future volatility. In equity markets, this is unrealistic.
- No leverage effect. Empirically, negative returns tend to increase future volatility more than positive returns of the same magnitude (Black, 1976). Standard ARCH cannot capture this asymmetry. Extensions like EGARCH and GJR-GARCH were specifically designed to address it.

These limitations don't make ARCH useless. They make it a building block. Understanding where ARCH falls short is the clearest path to understanding why GARCH and its variants exist.

14.3 GARCH models and extensions

GARCH models expand on ARCH models by including past conditional variances in the current variance equation. This produces a more efficient representation of volatility processes, capturing long-term dependencies with fewer parameters and reducing overfitting risk.

Extensions like EGARCH and TGARCH address asymmetric effects in volatility, particularly the "leverage effect" in financial time series. These models allow negative and positive shocks to impact volatility differently, which improves accuracy in forecasting and risk management.

GARCH Models

GARCH model vs ARCH models

An ARCH model explains today's variance using only past squared residuals (past shocks). A GARCH model adds lagged conditional variances to that equation, so the model "remembers" its own recent variance estimates too.

The GARCH(p, q) model is written as:

$$\sigma_t^2 = \omega + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2$$

- q is the number of lagged squared residuals (the ARCH terms)
- p is the number of lagged conditional variances (the GARCH terms)

The most commonly used specification is GARCH(1,1), which includes just one ARCH lag and one GARCH lag. Even this simple version captures volatility clustering remarkably well in practice.

Why prefer GARCH over a pure ARCH model?

- Parsimony. A GARCH(1,1) can replicate the memory of a high-order ARCH model with only three parameters ($\omega, \alpha_1, \beta_1$) instead of dozens.
- Better long-term memory. The lagged variance term lets the model capture slow-decaying volatility patterns that a short-lag ARCH model would miss.
- Lower overfitting risk. Fewer parameters means more stable estimates, especially with limited data.

GARCH models with lag structures

Choosing lag orders (p, q):

You need to pick p and q that fit the data without overcomplicating the model. In practice, GARCH(1,1) is the default starting point. Move to higher orders only if diagnostics suggest the simpler model is inadequate.

Estimation methods:

- Maximum likelihood estimation (MLE) is the standard approach. You assume a distribution for the standardized residuals (typically Gaussian or Student's t) and maximize the corresponding log-likelihood.
- Quasi-maximum likelihood estimation (QMLE) is useful when the assumed distribution is wrong. QMLE still gives consistent parameter estimates under weaker assumptions, though standard errors need adjustment.

Model selection and diagnostics:

1. Fit candidate models with different (p, q) combinations.
2. Compare them using information criteria (AIC, BIC). Lower values indicate a better trade-off between fit and complexity.
3. Check the standardized residuals of your chosen model: - Ljung-Box test on squared standardized residuals to detect remaining serial correlation in variance. - ARCH-LM test to check whether any ARCH effects are left unexplained.
4. If these tests reject, consider increasing the lag order or switching to an extension model.

Volatility persistence in forecasting

Volatility persistence measures how long the effect of a shock to variance lingers. For a GARCH model, it's captured by the sum:

$$\sum_{i=1}^q \alpha_i + \sum_{j=1}^p \beta_j$$

For a GARCH(1,1), this simplifies to $\alpha_1 + \beta_1$.

- When this sum is close to 1, shocks to volatility decay very slowly. A spike in variance today will still influence variance estimates far into the future.
- When the sum equals 1 exactly, you have an Integrated GARCH (IGARCH) process, where shocks never fully die out and the unconditional variance is undefined.
- When the sum is well below 1, volatility reverts to its long-run level relatively quickly.

Implications for forecasting:

- High persistence means current volatility has a strong influence on future volatility, so short-horizon forecasts will differ substantially from the long-run average.
- Long-term forecasts converge to the unconditional variance $\frac{\omega}{1 - \alpha_1 - \beta_1}$, but convergence is slower when persistence is high.
- Confidence intervals for multi-step-ahead forecasts widen more when persistence is high, reflecting greater uncertainty.

GARCH Model Extensions

Extensions of GARCH models

Standard GARCH treats positive and negative shocks symmetrically: a +2% return shock and a -2% return shock produce the same effect on variance. In reality, financial markets often exhibit a leverage effect, where negative shocks increase volatility more than positive shocks of equal size. The extensions below address this.

EGARCH (Exponential GARCH)

EGARCH models the log of the conditional variance, which has two benefits: it naturally prevents negative variance estimates, and it allows asymmetry through a sign term.

$$\log(\sigma_t^2) = \omega + \sum_{i=1}^q \alpha_i \frac{|\varepsilon_{t-i}|}{\sigma_{t-i}} + \sum_{j=1}^p \gamma_j \frac{\varepsilon_{t-j}}{\sigma_{t-j}} + \sum_{k=1}^p \beta_k \log(\sigma_{t-k}^2)$$

- The γ_j coefficients capture asymmetry. If $\gamma_j < 0$, negative shocks increase volatility more than positive shocks.
- Because the equation is in logs, there's no need for non-negativity constraints on the parameters.

TGARCH / GJR-GARCH (Threshold GARCH)

These models introduce an indicator function that "switches on" an extra term when the shock is negative:

$$\sigma_t^2 = \omega + \sum_{i=1}^q (\alpha_i + \gamma_i I_{t-i}) \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2$$

where $I_{t-i} = 1$ if $\varepsilon_{t-i} < 0$, and 0 otherwise.

- For a positive shock, the impact on variance is α_i .
- For a negative shock, the impact is $\alpha_i + \gamma_i$. If $\gamma_i > 0$, negative shocks have a larger effect, confirming the leverage effect.
- The GJR-GARCH formulation is very similar to TGARCH, with minor differences in how the threshold term enters.

Why these extensions matter:

The leverage effect is well-documented in equity markets: stock price declines raise a firm's debt-to-equity ratio, increasing perceived risk and therefore volatility. Ignoring this asymmetry leads to biased volatility forecasts. These extensions are widely used in risk management applications such as portfolio optimization and option pricing, where accurate directional volatility estimates are critical.

Unit 15 – Time Series in Finance and Economics

15.1 Stock price and return analysis

Stock price analysis breaks down price movements into components like trend, seasonality, and cyclical fluctuations so you can better understand market behavior. Time series modeling takes this further by using techniques like ARIMA and GARCH to forecast future prices and quantify volatility, helping analysts manage risk more effectively.

Stock Price Time Series Analysis

Analysis of stock price trends

Stock prices aren't just random numbers bouncing around. You can decompose their movements into distinct components, each telling you something different about what's driving the price.

- Trend captures the long-term upward or downward direction (think bull vs. bear markets over years)
- Seasonality refers to recurring patterns at fixed intervals, like the tendency for prices to shift around quarterly earnings reports
- Cyclical fluctuations are tied to broader economic cycles such as expansions and recessions. Unlike seasonality, these don't have a fixed period.
- Irregularity is the noise from unexpected events: natural disasters, geopolitical shocks, surprise announcements

Visual analysis techniques help you spot these components before running any formal tests:

- Line plots show overall trends and patterns over time
- Candlestick charts display the open, high, low, and close prices for each trading period (day, week, or month), giving you a richer picture than a simple line
- Moving averages smooth out short-term noise to reveal the underlying trend. A 50-day moving average reacts faster to recent changes, while a 200-day moving average shows the broader direction. When a shorter moving average crosses above a longer one, traders often interpret that as a bullish signal.

Statistical analysis techniques let you quantify what you're seeing visually:

- Autocorrelation (ACF) measures the correlation between a stock price and its own past values at various lags (e.g., how correlated is today's price with yesterday's?)
- Partial autocorrelation (PACF) measures the correlation at a specific lag after removing the effects of all intermediate lags. This helps isolate the direct relationship between, say, today's price and the price two days ago.
- Augmented Dickey-Fuller (ADF) test checks whether the time series is stationary, meaning its statistical properties (mean, variance) stay roughly constant over time. Raw stock prices almost always fail this test, which is one reason we often work with returns instead.

Anomaly detection methods flag unusual behavior:

- Outlier detection identifies observations that deviate significantly from the norm, such as sudden spikes or crashes
- Change point detection finds abrupt shifts in the series' properties, like a structural break where the market enters a fundamentally different regime

Calculation of stock returns

Raw stock prices are hard to model directly because they're non-stationary. Converting prices to returns solves this problem and makes different stocks comparable regardless of their price level.

Simple returns calculate the percentage change over one period:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

A positive value means the price went up; negative means it went down. If a stock goes from \$100 to \$105, the simple return is 0.05, or 5%.

Log returns give you a continuously compounded measure:

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

Log returns have two big advantages for time series work. First, they're additive across time: to get the total log return over multiple periods, you just sum the individual log returns. Second, they're more symmetric than simple returns, which makes them better suited for statistical models that assume normality. For small returns, simple and log returns are nearly identical.

Adjusted returns account for dividends, stock splits, and other corporate actions:

$$R_{adj,t} = \frac{P_t + D_t - P_{t-1}}{P_{t-1}}$$

Here D_t represents dividends paid during the period. Adjusted returns give a more accurate picture of the total return an investor actually receives, not just the price change.

Time Series Modeling and Forecasting

Time series models for forecasting

ARIMA models (Autoregressive Integrated Moving Average) capture linear dependencies in stock price data. Fitting one involves three steps:

1. Identify the model order. Use the ACF and PACF plots along with information criteria (AIC, BIC) to choose the number of AR terms (p), differencing steps (d), and MA terms (q). The differencing step is what makes a non-stationary price series stationary.
2. Estimate parameters. Fit the model using maximum likelihood or least squares estimation.
3. Diagnose the fit. Check that the residuals look like white noise: they should be approximately normal, have constant variance (homoscedasticity), and show no significant autocorrelation.

GARCH models (Generalized Autoregressive Conditional Heteroskedasticity) address something ARIMA can't: time-varying volatility. Stock returns tend to show volatility clustering, where periods of high volatility bunch together. GARCH models capture this pattern.

1. Identify the order of the ARCH (q) and GARCH (p) terms using information criteria.
2. Estimate parameters via maximum likelihood.

3. Diagnose the model by checking that the standardized residuals are approximately normal and independent.

GARCH models are widely used in risk management (e.g., calculating Value at Risk) and option pricing (estimating implied volatility).

Evaluation of forecasting models

No model is useful unless you can measure how well it actually predicts. Several metrics quantify forecast accuracy:

- MAE (Mean Absolute Error) gives the average absolute difference between forecasts and actual values. Easy to interpret.
- MSE (Mean Squared Error) squares the errors before averaging, so large errors get penalized more heavily.
- RMSE (Root Mean Squared Error) is the square root of MSE, putting the error back in the same units as the stock price.
- MAPE (Mean Absolute Percentage Error) expresses errors as percentages, making it easier to compare across stocks with different price levels.

Validation techniques test whether your model generalizes beyond the data it was trained on:

- Train-test split divides your data chronologically (e.g., 80% for training, 20% for testing). For time series, you always train on earlier data and test on later data to avoid look-ahead bias.
- Rolling window moves the training window forward through time, refitting the model and generating forecasts at each step. This simulates how the model would perform in real-time use.
- Time series cross-validation is similar to k-fold cross-validation but respects the temporal order of the data, expanding the training set at each fold.

Benchmark comparisons put your model's performance in context:

- Always compare against a naive forecast like the random walk (tomorrow's price equals today's price) or a simple historical average. If your model can't beat these, it's not adding value.
- Use formal statistical tests like the Diebold-Mariano test to check whether differences in predictive accuracy between two models are statistically significant, rather than just eyeballing the error metrics.

15.2 Exchange rate modeling

Exchange rates are influenced by various factors like interest rates, inflation, and economic growth. Understanding these factors is crucial for predicting currency movements and their impact on global trade and investments.

Time series models like ARIMA and VAR help forecast exchange rates and analyze their relationships with other economic variables. These models offer insights into short-term fluctuations and long-term equilibrium, aiding decision-making in finance and policy.

Factors Influencing Exchange Rates and Time Series Modeling

Factors in exchange rate movements

- Interest rates

- Higher interest rates attract foreign investment leading to increased demand for the domestic currency and appreciation (e.g., when the Federal Reserve raises interest rates)
- Lower interest rates may lead to capital outflows and currency depreciation as investors seek higher returns elsewhere (e.g., when the European Central Bank maintains low interest rates)
- Inflation
 - Higher inflation rates can lead to currency depreciation as the purchasing power of the currency decreases (e.g., Venezuela's hyperinflation)
 - Countries with lower inflation rates tend to have more stable and attractive currencies (e.g., Switzerland's low inflation and strong Swiss franc)
- Economic growth
 - Stronger economic growth often leads to currency appreciation due to increased foreign investment and confidence in the economy (e.g., China's rapid economic growth and appreciation of the yuan)
 - Weak economic growth or recession may result in currency depreciation as investors lose confidence and capital flows out (e.g., Greece's economic crisis and depreciation of the euro)

Time series models for forecasting

- ARIMA (Autoregressive Integrated Moving Average)
 - Univariate model that captures the autocorrelation structure of a single time series
 - Useful for short-term forecasting of exchange rates (e.g., predicting the USD/EUR exchange rate for the next month)
 - Consists of three components:
 1. Autoregressive (AR): Models the relationship between an observation and a certain number of lagged observations
 2. Differencing (I): Removes trend and seasonality by differencing the series
 3. Moving Average (MA): Models the relationship between an observation and a residual error from a moving average model applied to lagged observations
- VAR (Vector Autoregression)
 - Multivariate model that captures the dynamic relationships among multiple time series variables
 - Useful for analyzing the interdependencies between exchange rates and other economic variables (e.g., studying the relationship between exchange rates, interest rates, and inflation)
 - Treats all variables as endogenous and allows for feedback effects, meaning each variable can influence and be influenced by other variables
- Cointegration
 - Concept of long-run equilibrium relationship between non-stationary time series
 - Useful for identifying long-term relationships between exchange rates and other economic variables (e.g., the relationship between exchange rates and purchasing power parity)
 - Enables the estimation of error correction models (ECM) to capture short-term dynamics and adjustment towards long-run equilibrium

Interpretation, Limitations, and Challenges of Exchange Rate Modeling

Implications of exchange rate analysis

- Forecasting accuracy
- Evaluate the performance of exchange rate models based on measures such as mean squared error (MSE) or mean absolute percentage error (MAPE)
- Accurate forecasts can help businesses and investors make informed decisions regarding international transactions and risk management (e.g., hedging strategies for multinational corporations)
- Economic implications
- Appreciate the impact of exchange rate movements on international trade competitiveness (e.g., a weaker currency can boost exports but make imports more expensive)
- Understand how exchange rate fluctuations affect the profitability of multinational corporations and the attractiveness of foreign investments (e.g., a stronger currency can reduce the value of foreign earnings when converted back to the domestic currency)
- Policy implications
- Recognize the role of exchange rate modeling in informing monetary and fiscal policy decisions (e.g., central banks using exchange rate forecasts to guide interest rate decisions)
- Consider how exchange rate forecasts can influence central bank interventions and currency stabilization efforts (e.g., the People's Bank of China managing the yuan's exchange rate)

Limitations of exchange rate modeling

- Model assumptions
- Be aware of the simplifying assumptions underlying exchange rate models, such as market efficiency and rationality, which may not always hold in reality
- Recognize that models may not capture all relevant factors influencing exchange rates, such as political events or market sentiment (e.g., the impact of Brexit on the British pound)
- Data limitations
- Understand the challenges posed by data availability, frequency, and quality when modeling exchange rates (e.g., the difficulty of obtaining reliable data for emerging market currencies)
- Consider the impact of data revisions and structural breaks on model performance (e.g., the COVID-19 pandemic causing significant disruptions to economic data)
- Market complexity
- Appreciate the complexity and interconnectedness of global financial markets, which can make exchange rate modeling challenging
- Recognize that exchange rates are influenced by a wide range of economic, political, and psychological factors (e.g., the impact of geopolitical tensions on currency markets)
- Structural changes

- Be aware of the potential for structural changes in the economy or financial system that can affect exchange rate dynamics (e.g., the introduction of a new currency or the adoption of a different exchange rate regime)
- Understand that models may need to be adapted or re-estimated in the face of changing market conditions or policy regimes to maintain their relevance and accuracy

15.3 Economic indicators and business cycle analysis

Economic indicators are vital tools for understanding and predicting economic trends. These measures, like GDP, inflation, and unemployment, provide insights into the overall health of an economy and help guide decision-making for businesses, policymakers, and individuals.

Time series analysis techniques, such as decomposition and forecasting models, allow us to break down these indicators into their core components. By examining trends, seasonal patterns, and cyclical fluctuations, we can better interpret economic data and make more informed predictions about future economic conditions.

Economic Indicators

Key economic indicators

- Gross Domestic Product (GDP) measures total value of goods and services produced within a country's borders, gauges overall economic health and size, identifies periods of expansion and contraction
- Inflation measures rate at which general price level of goods and services rises, indicates changes in purchasing power and value of money (Consumer Price Index (CPI), Producer Price Index (PPI))
- Unemployment measures percentage of labor force jobless but actively seeking employment, provides insights into labor market and economic health (frictional, structural, cyclical unemployment)
- Consumer Confidence measures degree of consumer optimism about the economy and personal finances, influences consumer spending (major component of GDP), examples include Consumer Confidence Index (CCI) and University of Michigan Consumer Sentiment Index

Time series decomposition techniques

- Time series decomposition separates a series into constituent components (trend, seasonal, cyclical), identifies underlying patterns and isolates specific influencing factors
- Trend component represents long-term direction of series (linear or non-linear), estimated using moving averages and polynomial regression
- Seasonal component captures regular, predictable patterns within a year (increased retail sales during holidays, reduced winter construction activity), removed using seasonal adjustment methods (X-13ARIMA-SEATS)
- Cyclical component represents medium-term fluctuations related to business cycle, characterized by alternating periods of expansion and contraction, isolated by removing trend and seasonal components

Relationships between economic indicators

- Cross-correlation measures similarity between two series as function of lag, identifies strength and direction of relationship, determines lead or lag

- Lead-lag analysis examines timing of turning points in different indicators 1. Leading indicators change before overall economy, provide early signals of future activity (stock prices, building permits, new orders for manufactured goods) 2. Lagging indicators change after overall economy, confirm patterns in other indicators (unemployment rate, average duration of unemployment, inventory-to-sales ratio) 3. Coincident indicators change at approximately same time as overall economy (personal income, industrial production, manufacturing and trade sales)

Forecasting models for economic indicators

- Vector Autoregressive (VAR) models are multivariate models capturing dynamic relationships among multiple variables, each modeled as function of own past values and past values of other variables, used for forecasting and analyzing impact of shocks (impulse response functions, variance decompositions)
- State-space models represent series as combination of unobserved components (trend, cycle, irregular), consist of measurement equation relating observed data to unobserved components and transition equation describing evolution of unobserved components over time (Harvey-Clark model, Kalman filter), used for forecasting, signal extraction, estimating unobserved variables (output gap)

Implications of business cycle analysis

- Business cycle phases: expansion (increasing GDP, low unemployment, rising inflation), peak (highest point, end of expansion), contraction (decreasing GDP, rising unemployment, falling inflation), trough (lowest point, end of contraction)
- Economic policy implications: monetary policy (central banks adjust interest rates or use quantitative easing to stimulate economy during contractions or control inflation during expansions), fiscal policy (governments increase spending or reduce taxes to stimulate economy during contractions or reduce spending and raise taxes to cool economy during expansions)
- Decision-making implications: businesses adjust investment, production, hiring based on current and expected state of business cycle; investors adjust portfolio allocations based on expected performance of asset classes throughout cycle; consumers adjust spending and saving based on expectations about future economic conditions and job security

Unit 16 – Environmental Science Applications

16.1 Climate data analysis

Climate data analysis applies time series techniques to understand long-term trends and patterns in environmental datasets. These methods let researchers quantify warming trends, detect cyclical patterns like El Niño, and forecast future climate conditions, all of which feed directly into policy decisions and adaptation planning.

Climate Data Analysis Techniques

Time series analysis of climate trends

Three main approaches help you identify and measure trends in climate data: regression-based methods, smoothing techniques, and decomposition.

Trend Analysis

- Linear regression fits a straight line to the data using least squares. The slope tells you the direction and magnitude of the trend: a positive slope indicates warming, a negative slope indicates cooling.
- Mann-Kendall test is a non-parametric test for monotonic trends (consistent increase or decrease over time). Because it doesn't assume normality, it handles the outliers and extreme values common in climate data.
- Sen's slope estimator calculates the median slope among all pairs of data points. Taking the median rather than the mean makes it resistant to outliers, giving a more stable trend estimate than ordinary regression.

Smoothing Techniques

- Moving average reduces short-term fluctuations to highlight long-term trends. A 10-year moving average is common for climate data. Larger windows produce smoother curves; smaller windows preserve more detail.
- Exponential smoothing assigns exponentially decreasing weights to older observations, so recent data has more influence. This works well for data without a clear trend or seasonal pattern, such as temperature anomalies.

Decomposition Methods

Decomposition splits a time series into three components:

- T_t : the trend component (long-term direction)
- S_t : the seasonal component (recurring within-year pattern)
- R_t : the remainder/irregular component (unexplained variation)

The two standard forms are:

- Additive: $Y_t = T_t + S_t + R_t$

- Multiplicative: $Y_t = T_t \times S_t \times R_t$

Use additive decomposition when the seasonal swings stay roughly constant over time. Use multiplicative when the seasonal variation grows or shrinks proportionally with the trend.

Detection of climate data patterns

Once you've decomposed a series, several tools help you identify and characterize the cyclical patterns within it.

Seasonal decomposition isolates the seasonal component from the time series. Seasonal indices represent the average deviation from the trend for each season. For temperature data, summer months typically sit above the trend and winter months below.

Fourier analysis expresses the time series as a sum of sine and cosine waves. Each wave has a frequency and amplitude, so you can identify dominant cycles: annual, semi-annual, decadal, or others. This is particularly useful for detecting phenomena like the El Niño Southern Oscillation (ENSO), which operates on irregular multi-year cycles.

Autocorrelation function (ACF) measures the correlation between a time series and its own lagged values. Seasonal patterns show up as significant autocorrelation at seasonal lags. For example, January temperatures in consecutive years tend to be highly correlated.

Periodogram estimates spectral density, which is how variance is distributed across frequencies. Peaks in the periodogram point to dominant cycles. Temperature data, for instance, typically shows a strong peak at the annual frequency.

Fourier analysis and the periodogram are closely related. Fourier analysis decomposes the signal; the periodogram visualizes where the variance concentrates across frequencies.

Statistical significance of climate trends

Detecting a trend visually isn't enough. You need to test whether it's statistically distinguishable from random noise.

Hypothesis testing follows the standard framework:

1. Set up the null hypothesis (H_0): no significant trend (slope equals zero).
2. Set up the alternative hypothesis (H_a): a significant trend exists (slope is non-zero).
3. Compute the p-value, which is the probability of observing data this extreme if H_0 were true.
4. Reject H_0 if the p-value falls below your significance level (typically 0.05).

Confidence intervals quantify uncertainty around the estimated trend. A 95% confidence interval gives a range of plausible values for the true trend. Narrow intervals mean more precise estimates; wide intervals signal more uncertainty, often due to high variability or limited data.

Bootstrap resampling is a non-parametric alternative that doesn't assume a specific distribution:

1. Resample the data with replacement to create many new datasets.
2. Calculate the trend for each resample.
3. Use the distribution of resampled trends to build confidence intervals (e.g., the 2.5th and 97.5th percentiles for a 95% interval).

This approach is especially valuable for climate data, where the underlying distribution may not be well-behaved.

Climate forecasting models from historical data

ARIMA models (Autoregressive Integrated Moving Average) combine three components: autoregressive (AR) terms that use past values, differencing (I) to remove trends and achieve stationarity, and moving average (MA) terms that use past forecast errors. Standard ARIMA works well for non-seasonal series like annual global temperature anomalies. Model selection typically relies on information criteria such as AIC or BIC, which balance model complexity against goodness of fit.

SARIMA models (Seasonal ARIMA) extend ARIMA by adding seasonal AR, differencing, and MA terms. This lets them capture both the long-term trend and recurring seasonal patterns. Monthly sea surface temperature data is a classic use case.

Exponential smoothing models offer a complementary approach:

- Holt-Winters method handles data with both trend and seasonality. You choose additive seasonality when seasonal swings are constant in magnitude, or multiplicative when they scale with the level.
- Damped trend models gradually flatten the trend over the forecast horizon. These are useful for long-term climate projections where assuming the current trend continues indefinitely would be unrealistic.

Model evaluation is critical for any forecasting approach:

1. Split the data into training and testing sets (e.g., 80/20).
2. Fit the model on the training set and generate forecasts for the test period.
3. Assess accuracy using metrics like RMSE (root mean squared error), MAE (mean absolute error), or MAPE (mean absolute percentage error).
4. For more robust evaluation, use cross-validation techniques like rolling origin forecasting, which repeatedly moves the train-test boundary forward through time. This reduces the risk that your results depend on one particular split.

16.2 Air quality modeling

Air quality modeling combines emission sources, atmospheric processes, and chemical transformations to predict pollutant levels. Key components include identifying sources, understanding transport and dispersion, and accounting for chemical reactions and meteorological factors that influence air quality.

Time series analysis plays a crucial role in air quality monitoring and prediction. It involves data collection, decomposition of pollutant concentrations, modeling techniques like ARIMA, and forecasting. This approach helps understand trends, seasonal patterns, and the impact of interventions on air quality.

Key Concepts in Air Quality Modeling

Components of air quality modeling

- Emission sources encompass anthropogenic activities (industrial processes, transportation, residential heating) and natural phenomena (wildfires, volcanic eruptions, dust storms)
- Atmospheric transport and dispersion involve advection (horizontal movement of pollutants by wind), diffusion (spreading due to turbulent mixing), and deposition (removal through dry or wet processes)

- Chemical transformations distinguish between primary pollutants emitted directly from sources (CO, SO₂, NO_x) and secondary pollutants formed through atmospheric reactions (O₃, PM_{2.5})
- Meteorological factors play a crucial role, with wind speed and direction influencing transport and dispersion, temperature affecting reaction rates and vertical mixing, humidity impacting secondary pollutant formation, and atmospheric stability determining vertical mixing and dispersion

Time Series Analysis in Air Quality Monitoring and Prediction

Time series for pollutant concentrations

- Data collection and preprocessing involve continuous monitoring using ground-based stations or satellite observations, followed by quality control and data cleaning to remove outliers and fill missing values
- Time series decomposition separates the data into trend (long-term changes), seasonal (recurring patterns related to meteorology or human activities), cyclical (variations longer than one year), and irregular (random fluctuations or unexpected events) components
- Time series modeling techniques include autoregressive (AR) models predicting future values based on past observations, moving average (MA) models accounting for past forecast errors, and autoregressive integrated moving average (ARIMA) models combining AR and MA components with differencing for non-stationary data
- Forecasting and model evaluation generate short-term and long-term predictions using trained models and assess performance using metrics like mean absolute error (MAE), root mean square error (RMSE), and coefficient of determination (R^2)

Meteorological factors in air quality

- Correlation analysis calculates coefficients between pollutant concentrations and meteorological variables to identify significant relationships (temperature and ozone levels)
- Regression models quantify the impact of meteorological factors on pollutant concentrations, including interaction terms to capture combined effects
- Granger causality tests assess causal relationships, determining if changes in meteorological variables lead to changes in air quality

Statistical models for pollution control

- Intervention analysis uses dummy variables or step functions to represent the implementation of control policies or regulations in the time series model, estimating the magnitude and significance of the intervention effect on pollutant concentrations
- Difference-in-differences (DID) approach compares changes in air quality between a treatment group (affected by the control measure) and a control group (unaffected), accounting for common trends and isolating the causal impact
- Counterfactual analysis constructs a scenario representing the expected air quality without the control measure and compares observed pollutant levels with the counterfactual to quantify the intervention's effectiveness
- Policy evaluation metrics calculate the percentage reduction in pollutant concentrations attributable to the control measure and estimate the associated health and economic benefits

16.3 Hydrological time series analysis

Hydrological Time Series Analysis

Hydrological time series analysis applies the core tools of time series modeling to water-related data: streamflow, water levels, precipitation, and similar measurements. Understanding these patterns is essential for predicting water availability, managing flood risk, and planning infrastructure. Climate change and land use shifts are actively reshaping hydrological systems, making trend detection and scenario modeling more important than ever.

Analysis of Streamflow Data

Before you can model or forecast anything, you need clean data and a solid understanding of what's in it. Hydrological time series analysis typically follows three stages: preprocessing, exploratory analysis, and pattern identification.

Preprocessing ensures your data is reliable before any modeling begins:

- Handle missing values and outliers using interpolation or statistical methods (mean imputation, regression imputation, or more advanced gap-filling techniques)
- Apply standardization or normalization so data from different stations or measurement scales can be compared (z-score standardization, min-max scaling)

Exploratory data analysis helps you understand the structure of your data visually and numerically:

- Inspect time series plots (line plots, scatter plots) to spot patterns, anomalies, and apparent trends
- Calculate summary statistics to characterize the distribution: mean, median, standard deviation, skewness, and kurtosis

Pattern identification uses formal time series tools to uncover structure:

- Autocorrelation (ACF) and partial autocorrelation (PACF) functions reveal dependencies and repeating patterns in the data, such as seasonality or persistence from one time step to the next
- Spectral analysis detects periodic components and their dominant frequencies. For streamflow data, you'll often find strong annual cycles and sometimes sub-annual (e.g., monthly) patterns
- Trend analysis determines whether water levels or streamflow are systematically increasing or decreasing over time

Trends in Hydrological Data

Most hydrological time series contain multiple overlapping signals. Decomposition separates these into interpretable components so you can study each one individually.

Additive vs. multiplicative decomposition:

- Additive model: $Y_t = T_t + S_t + R_t$. Use this when the seasonal fluctuations stay roughly constant in magnitude over time.
- Multiplicative model: $Y_t = T_t \times S_t \times R_t$. Use this when seasonal swings grow or shrink proportionally with the level of the series.

In both cases:

- T_t is the trend component, capturing long-term direction (gradual increase or decrease in streamflow, for example)
- S_t is the seasonal component, capturing recurring patterns within a fixed period (annual snowmelt peaks, monsoon cycles)
- R_t is the residual component, the leftover fluctuations after trend and seasonality are removed. Analyzing residuals helps you judge whether your decomposition captured the important structure.

Estimating trends can be done several ways:

- Linear or nonlinear regression to fit a trend line (linear, quadratic, or other functional forms)
- Moving averages and smoothing techniques (simple moving average, exponential smoothing) to filter out noise and reveal the underlying direction

SARIMA models handle both seasonal and non-seasonal patterns in a single framework. The notation is $\text{SARIMA}(p, d, q)(P, D, Q)^m$, where:

- p, d, q are the non-seasonal autoregressive order, differencing order, and moving average order
- P, D, Q are the seasonal counterparts of those same three parameters
- m is the number of periods per season (e.g., 12 for monthly data with an annual cycle, 4 for quarterly data)

The non-seasonal parameters capture short-term dependencies, while the seasonal parameters capture patterns that repeat every m time steps. For example, a SARIMA model on monthly streamflow data would use $m = 12$ to account for the annual cycle.

Forecasting of Water Availability

Forecasting water availability means choosing a model, fitting it to historical data, and then evaluating how well it predicts unseen observations. Several model families are commonly used.

ARIMA models are the standard statistical approach. The $\text{ARIMA}(p, d, q)$ notation means:

- p : number of lagged values of the variable used as predictors (autoregressive terms)
- d : number of times the series is differenced to achieve stationarity
- q : number of lagged forecast errors used as predictors (moving average terms)

Building an ARIMA model follows these steps:

1. Check for stationarity (using plots and tests like the Augmented Dickey-Fuller test). If the series isn't stationary, difference it until it is. The number of differences becomes d .
2. Examine the ACF and PACF of the differenced series to identify candidate values for p and q .
3. Fit candidate models and compare them using information criteria (AIC, BIC). Lower values indicate a better balance of fit and parsimony.
4. Estimate parameters via maximum likelihood or least squares.
5. Run diagnostic checks on the residuals: they should resemble white noise (no remaining autocorrelation, approximately normal distribution).

Exponential smoothing methods offer a simpler alternative, chosen based on the data's characteristics:

- Simple exponential smoothing for data with no trend or seasonality
- Holt's linear trend method for data with a trend but no seasonality
- Holt-Winters' seasonal method for data with both trend and seasonality

Machine learning approaches can capture nonlinear relationships that statistical models may miss:

- Artificial neural networks (ANNs) learn complex patterns from the data
- Support vector machines (SVMs) perform regression-based forecasting
- Random forests, an ensemble method, can improve accuracy and robustness by averaging many decision trees

Model evaluation and selection is critical regardless of which approach you use:

- Split data using cross-validation strategies suited to time series (rolling window validation is preferred over random k-fold, since time order matters)
- Calculate error metrics to quantify forecast accuracy:
 - RMSE (root mean squared error) penalizes large errors heavily
 - MAE (mean absolute error) gives equal weight to all errors
 - MAPE (mean absolute percentage error) expresses error as a percentage, making it easier to interpret across different scales
- Compare models using AIC and BIC, which penalize model complexity to guard against overfitting

Impact Assessment and Climate Change

Climate Impact on Hydrological Processes

Climate change and land use change both alter how water moves through a landscape. A key challenge is figuring out which factor is responsible for observed changes, and how they interact.

Detecting climate change impacts:

- Trend analysis of temperature and precipitation records reveals long-term shifts (warming trends, altered rainfall patterns)
- Changes in hydrological regimes show up as shifts in timing or magnitude of streamflow. Earlier snowmelt, more frequent floods, and longer dry spells are common examples.

Assessing land use change impacts:

- Urbanization and deforestation increase surface runoff and reduce infiltration, which lowers groundwater recharge
- Agricultural practices like irrigation and water abstraction can reduce streamflow and increase nutrient loads in waterways

Attributing changes to specific causes requires statistical tools that can separate overlapping influences:

- Multiple regression and principal component analysis help isolate the contributions of climate vs. land use
- Multivariate time series techniques capture relationships between variables:

- Cointegration and error correction models identify long-run equilibrium relationships (e.g., between water demand and supply, or groundwater levels and pumping rates) along with short-term dynamics
- Granger causality tests assess whether changes in one variable systematically precede changes in another (e.g., does land use change predict subsequent changes in streamflow?)

Scenario analysis and projections use models to explore possible futures:

- Climate model outputs (under scenarios like RCP4.5 or RCP8.5) serve as inputs to hydrological models, producing projections of future water availability and flood risk
- Different land use scenarios (afforestation, sustainable agriculture) can be modeled to estimate their hydrological effects

Adaptation and mitigation strategies translate analysis into action:

- Water resource management practices such as conservation programs and demand management help cope with shifting conditions
- Land use planning measures like zoning regulations and riparian buffer zones can reduce the negative hydrological effects of development