UNIVERSITY OF MARYLAND

MSML610: Advanced Machine Learning

Machine Learning on Time Series

Instructor: GP Saggese, PhD - gsaggese@umd.edu

References:

Time Series

- **Time Series**
  - Basic definition
  - Time series operators
  - Time series decomposition
- Classical Methods
- Advanced and Modern Approaches
- Special techniques for time series modeling

Basic definition

- Time Series
    - **Basic definition**
    - Time series operators
    - Time series decomposition
- Classical Methods
- Advanced and Modern Approaches
- Special techniques for time series modeling

Time Series

- A **time series** is a sequence of observations over time, e.g.,
  - Finance: Hourly stock prices
  - Web Analytics: Number of active users on a site sampled at regular intervals
  - Manufacturing: Sensor data from machinery (e.g., temperature or vibration) collected over time for predictive maintenance
  - Weather: Daily temperature measurements
  - Energy: Daily electricity usage of a household
- Time series are needed only for things that change over time
  - Everything in the real world (besides mathematical objects) changes over time!
- Goal: understand patterns and predict future values
- A time series is modeled as a random process, i.e., a sequence of random variables indexed by time:

$$\{Y_t\}_{t=-\infty}^{\infty}$$

  - Can be continuous or discrete
  - Often consider data equi-spaced in time
  - The time dimension matters since random variables exhibit dependence

Time Series Visualization and Exploration

- **Visualization:**
  - Guides preprocessing choices
  - Helps form hypotheses before modeling
- Distinguish between underlying structure and randomness
  - **Trend**: long-term increase or decrease
  - **Seasonality**: repeating patterns at regular intervals
  - **Noise**: random fluctuations
- **Line plots** show raw data over time, e.g.,
  - Trend presence
  - Outliers or abrupt changes
- **Seasonal plots** reveal periodic patterns
  - E.g., plot monthly sales to find yearly seasonality
- **Autocorrelation plots (ACF)** detect repeating structures

j-lag autocovariance

- The j-lag autocovariance of a time series $\{Y_t\}$ is:

$$\mathrm{Cov}(Y_t, Y_{t-j}) \stackrel{def}{=} \mathbb{E}[(Y_t - \mathbb{E}[Y_t])(Y_{t-j} - \mathbb{E}[Y_{t-j}])]$$

  - Covariance of a random variable and the variable j samples before
- The j-lag autocorrelation of a time series $\{Y_t\}$ is:

$$\rho(Y_t, Y_{t-j}) = \mathrm{Corr}(Y_t, Y_{t-j}) \stackrel{def}{=} \frac{\mathrm{Cov}(Y_t, Y_{t-j})}{\sqrt{\mathbb{V}[Y_t]\mathbb{V}[Y_{t-j}]}}$$

  - Measures strength and direction of the linear relationship between samples
  - Scale-free

Stationarity

- A time series $\{Y_t\}$ is **stationary** if some properties (e.g., mean, variance, autocorrelation structure) do not change over time, i.e., they are unchanged by shifts in time
  - Stationarity is analogous to IID sampling for random variables
- Time series are rarely stationary
  - Stationarity is often an approximation/simplification of reality
- **Why important:**
  - Many models (e.g., ARIMA) assume stationarity
  - E.g., raw stock prices are non-stationary, returns often are
- **Tests for stationarity:**
  - ADF Test (Augmented Dickey-Fuller): tests for unit root
  - KPSS Test: tests for trend stationarity

Strictly stationarity: definition

- A time series $\{Y_t\}$ is **strictly stationary** iff for any any set of $r > 0$ indices $t_1, t_2, ..., t_r < t$, the joint distribution of $(Y_{t_1}, Y_{t_2}, ..., Y_{t_r})$ depends only on the differences $t_1 - t_2, ..., t_1 - t_r$
  - E.g., (Y_1, Y_5) has the same joint distribution as (Y_{12}, Y_{16})
  - E.g., (Y_1, Y_2, Y_3) has the same joint distribution as (Y_3, Y_4, Y_5)
- Intuition:
  - The data (i.e., joint probability of any set of observations) is invariant when we shift it in time
  - Only the distances in time matter
- If $\{Y_t\}$ is strictly stationary:
  - All moments (e.g., mean, variance) of Y_t don't depend on t
  - Any statistics between lags of the time series depend only on the difference in time between lags

Weakly stationarity: definition

- Weakly stationarity requires weaker assumptions than for strictly stationary process:
  1. The mean is constant over time: $\mathbb{E}[Y_t] = \mu \ \ \forall t$
  2. The variance is constant over time: $\mathbb{V}[Y_t] = \sigma^2 \ \ \forall t$
  3. The j-lag autocovariance $\text{Cov}(Y_t, Y_{t-j})$ depends on distance between lags j but not on t: $\text{Cov}(Y_t, Y_{t-j}) = \gamma_j$
- In practice, there is a constraint only on:
  - the joint distribution of 2 time indices
  - first and second moments
- **Intuition**
  - No trend (mean is constant)
  - Variations around the mean have constant amplitude (variance is constant)
  - Consistent wiggling (random patterns look the same)

Auto-Correlation Function (ACF)

- Auto-correlation function is a graphical representation of the *i*-lag autocorrelation of a time series
- It is a plot of the correlation coefficient of a time series with its own lagged values
  - Ideally, plot also the uncertainty of the coefficients
- Partial Auto-Correlation Function (PACF) is like ACF but controls for the values of the time series at all shorter lags
  - The partial autocorrelation at lag k:

$$\alpha(k) = \mathrm{Corr}(Y_t - Proj_{t,k}(Y_t), Y_{t-k} - Proj_{t,k}(Y_{t,k}))$$

where $Proj_{t,k}(x)$ is the projection of x onto the space spanned by $(x_t, ..., x_{t-k+1})$

Transformation of a time series

- Any deterministic transformation $g()$ of a strictly (weakly) stationary process $\{Y_t\}$ is also strictly (weakly) stationary
- Sometimes there is a transformation that makes the process stationary, e.g.,
  - Detrending
  - Differencing (integer or fractional)
- **Log transformations** stabilize variance
  - Useful when data grows exponentially
- **Differencing** removes trend and makes series stationary
  - First difference: $y_t' = y_t - y_{t-1}$
- **Power transformations** (e.g., square root) can reduce skewness
- Detrending techniques:
  - Subtract a fitted trend line
  - Apply moving average smoothing

Time series operators

- Time Series
  - Basic definition
  - **Time series operators**
  - Time series decomposition
- Classical Methods
- Advanced and Modern Approaches
- Special techniques for time series modeling

Time series operators

- Time series operators $f(\cdot)$ (e.g., lag, difference) operates on a time series $\{X_t\}$ to generate another time series $\{Y_t\}$:

$$\{Y_t\} = f(\{X_t\})$$

Lag operator

- Given a time series $\{X_t\}$, the lag operator $L(\cdot)$ generates the time series:

$$Y_t = LX_t = X_{t-1}$$

- Aka "shift back", backshift, delay
- Intuition of lagging a time series:

$$Y_t = LX_t = X_{t-1}$$

  the t (e.g., today) element of the new time series is the $t-1$ (yesterday) element of the old time series, i.e., it delays the time series

Lag operator: positive sign

- The "normal" direction (i.e., with positive delay) is delaying / lagging
    - It is a positive sign since we are not snooping in the future
- This is the same convention of `pd.shift()`
- When using a variable function of time, it corresponds to $x(t-a)$ with $a > 0$

Shifting backwards

- When we shift backwards (aka lag) `df.shift(n>0)`, we move a value from the past to today

```
date        val    val.shift(2)
2016-03-10  0      nan
2016-03-11  1      nan
2016-03-14  2      0
2016-03-15  3      1
2016-03-16  4      2
```

- This is equivalent to “shifting down” a time series ordered by increasing dates
- The values at the beginning of the period are not available since they require data before the period of interest

Lead operator

- It is accomplished by:
$$Y_t = L^{-1} X_t = X_{t+1}$$
- Aka “shift forward”
- When using a variable function of time, the transformation is like $x(t+2)$ since the value today $x(0)$ is the value computed in the future $x(2)$
- MEM: we use a negative number in `df.shift(-2)` and in $x(t-a)$ with $a < 0$ which is ominous sign of snooping in the future

Shifting forward

- When we shift forward (aka lead) `df.shift(n<0)`, we move a value from the future (i.e., a value computed n periods in the future) to today

```
date        val    val.shift(-2)
2016-03-10  0      2
2016-03-11  1      3
2016-03-14  2      4
2016-03-15  3      nan
2016-03-16  4      nan
```

- This is equivalent to "shifting up" a time series ordered in the usual way (by increasing dates)
- A consequence is that:
    - Some values at the end of the period won't be available since they would have been computed after the period of interest is over
    - Some values computed at the beginning of the period will be discarded

Shifting more than one time step

- We can shift more than one lag with:

$$L^k X_t = X_{t-k}$$

$$L^{-k} X_t = X_{t+k}$$

Difference operator

- The first difference of a time series is defined as the time series:

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

  i.e., the time series that is the difference between the original time series and its lagged version

Difference operator in terms of lag operator

- The first difference can be written in terms of lag operator as the time series:

$$\Delta X_t = (1 - L)X_t$$

Second difference operator

- The second difference is defined as:

$$\Delta^2 X_t = \Delta(\Delta X_t)$$

- Developing:

$$Y_t = \Delta X_t = X_t - X_{t-1}$$

and

$$\begin{aligned} Z_t &= Y_t - Y_{t-1} \\ &= X_t - X_{t-1} - (X_{t-1} - X_{t-2}) \\ &= X_t - 2X_{t-1} + X_{t-2} \end{aligned}$$

- Note that this is not the difference $Y_t - Y_{t-2}$

Second difference operator in terms of lag operator

$$\Delta^2 X_t = (1 - L)^2 X_t$$

N-th difference operator

- The n-th difference operator is defined:

$$\Delta^n X_t = (1 - L)^n X_t$$

Differencing: intuition

- Differencing means computing the difference between consecutive observations:
$$Y_t = X_t - X_{t-1}$$
- This means removing the changes in the level of a time series, eliminating trend and seasonality, which stabilize the mean of the time series
- Differencing is a transformation applied to time series that can make it stationary

Differencing in terms of lag operator

- The (first order) difference can be written:

$$\Delta X_t = (1 - L)X_t$$

- The second order difference can be written:

$$\Delta^2 X_t = (1 - L)^2 X_t$$

- The n-th order difference can be written:

$$\Delta^n X_t = (1 - L)^n X_t$$

Time series decomposition

- Time Series
  - Basic definition
  - Time series operators
  - **Time series decomposition**
- Classical Methods
- Advanced and Modern Approaches
- Special techniques for time series modeling

Decomposition of Time Series

- Break time series into components:
  - **Trend**
    - Long-term increase or decrease in data
    - Can change direction over time
  - **Seasonality**
    - Affected by seasonal factors (e.g., time of day, day of week, month of year)
    - Fixed and known frequency
  - **Cycle**
    - Value rises and falls without fixed frequency
    - E.g., economic conditions exhibit cycles
  - **Residual (noise)**
- **Additive model**:
  - $y_t = \text{Trend}_t + \text{Seasonality}_t + \text{Residual}_t$
- The component can also be mixed in different ways (e.g., multiplicative)
- Visual decomposition helps in selecting the right model

Seasonality: example

- Consider antidiabetic drug sales
  - Sharp spike in January, dip in February, increase over the year
- Why?
  - In January, government subsidy makes it cost-effective to stockpile drugs
  - In February, dip occurs as people have already bought many drugs
  - Demand increases until December as people use their reserves
  - Then the cycle repeats next year

Cycle: example

- GDP moves up and down around its long-term growth trend
- There are different cycles:
  - Inventory: 3-5 years
  - Fixed investment: 7-11 years
  - Infrastructural investment: 15-25 years
  - Technological investment: 45-60 years

Seasonal plot

- Season plot allows visual inference and understand model structure
- Assume we know the periodicity of a signal (e.g., yearly periodicity of a monthly time series)
- Partition the time-series based on the periodicity:
  - E.g., for a time series with yearly periodicity, break the time series into yearly chunks
  - Plot each time series chunk on the same graph
  - Use a box plot if there are many observations
- **Questions:**
  - Do the data exhibit a seasonal pattern?
  - Is there a within-group pattern (e.g., Jan and July exhibit similar patterns)?
  - Are there outliers after accounting for seasonality?
  - Is the seasonality changing over time?

Seasonal sub-series plot

- The data for each season is collected together in a separate mini time plots
    - E.g., all the data points for Jan are plotted together as a time series

Seasonal differencing

- Instead of computing the difference between consecutive observations, take the difference between observations at the same point of consecutive periods
  - Useful for removing seasonal effects in time series data
  - E.g., for time series with yearly periodicity, take the Year-over-Year (YoY) difference
  - Helps in identifying underlying trends by eliminating seasonal fluctuations
- Particularly beneficial for data with strong seasonal patterns, such as retail sales or temperature data
  - E.g., if you have monthly sales data, compare January sales of one year to January sales of the next year to see the YoY change

Spectral plot

- Spectral plot estimates spectral density of a process from time samples of the signal
    - Detects periodicity
    - Identifies dominant frequencies
    - Analyzes power distribution over frequency
- E.g., in audio processing, a spectral plot identifies different frequencies in a sound recording, allowing for noise reduction or enhancement of certain frequencies

Classical Methods

- Time Series
- **Classical Methods**
    - Simple models for stochastic process
    - Autoregressive models
    - Moving average models
    - ARMA(p, q) process
    - ARIMA model
    - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

Simple models for stochastic process

- Time Series
- Classical Methods
    - **Simple models for stochastic process**
    - Autoregressive models
    - Moving average models
    - ARMA(p, q) process
    - ARIMA model
    - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

White noise process

- Defined as:

$$\{Y_t\} \sim \mathrm{WN}(0, \sigma^2)$$

- Each Y_t is a IID random variable at time t
  - Independent over time
  - Drawn from the same distribution (not necessarily Gaussian) $Y_t \sim$ IID from distribution F
  - With mean 0 and certain variance $\mathbb{E}[Y_t] = 0, \mathbb{V}[Y_t] = \sigma^2$
- **Key points:**
  - It's strictly stationary
  - $\{Y_t\}$ is uncorrelated over time
  - Variance σ^2 is constant for all t
  - $\mathrm{Cov}(Y_t, Y_{t-j}) = \gamma_j = 0$ for $j \neq 0$
- White noise is often used as a basic building block in time series analysis
  - E.g., if Y_t follows a Gaussian distribution (Gaussian white noise) $Y_t \sim \mathrm{IID}\ \mathcal{N}(0, \sigma^2)$
- It's called "white noise" because:
  - There is no pattern (i.e., it is "noise")

Deterministically trending process

- Defined as $Y_t = \beta_0 + \beta_1 t + \varepsilon_t$ where:
  - The noise is Gaussian: $\varepsilon_t \sim \mathrm{GWN}(0, \sigma_\varepsilon^2)$
  - The noise term is also called “innovation”, “error term”
- The mean $\mathbb{E}[Y_t] = \beta_0 + \beta_1 t$ depends on t
  - It is non-stationary in the mean

Random walk

- Defined as $Y_t = Y_{t-1} + \varepsilon_t$ where
  - The noise is Gaussian: $\varepsilon_t \sim \mathrm{GWN}(0, \sigma_\varepsilon^2)$
- It can be rewritten in terms of the noise terms doing a recursive substitution:

$$Y_t = Y_0 + \sum_{i=1}^{t} \varepsilon_i$$

- The mean is constant:

$$\mathbb{E}[Y_t] = \mathbb{E}[Y_0] = \mu$$

- The variance is:

$$\mathbb{V}[Y_t] = t\sigma_\varepsilon^2$$

  since all the covariances between innovations are 0
- It is non-stationary in the variance

Autoregressive models

- Time Series
- Classical Methods
    - Simple models for stochastic process
    - **Autoregressive models**
    - Moving average models
    - ARMA(p, q) process
    - ARIMA model
    - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

Autoregressive (AR) Models

- An AR model of order p predicts future values using past p values:

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

  where:

  - c is a constant
  - $\phi_1, \phi_2, \ldots, \phi_p$ are model parameters
  - y_t is the value at time t
  - $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$ are past values (lags)
  - ϵ_ti.i.d. $\mathcal{N}(0, \sigma^2)$ is white noise (random error term)

- E.g., predicting temperature today using temperature for past 3 days:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \phi_3 y_{t-3} + \epsilon_t$$

- AR models assume the time series is stationary:

  - Stationarity implies statistical properties of the series do not change over time
  - Partial Autocorrelation Function helps choose p
  - Model parameters are estimated using methods like Ordinary Least Squares (OLS) or Maximum Likelihood Estimation (MLE)

AR(1) process

- Aka "auto-regressive of order 1"
- AR(1) model is defined as:

$$Y_t = c + \phi Y_{t-1} + \varepsilon_t$$

  where the noise is IID Gaussian: $\varepsilon_t \sim \mathrm{GWN}(0, \sigma_\varepsilon^2)$

- MEM: AR(1) = autoregressive term + noise
- The representation:

$$Y_t = \phi Y_{t-1} + \varepsilon_t$$

  can be thought as a regression of Y_t against Y_{t-1}

- So it is regressive with respect to itself, i.e., "auto-regressive"

AR(1) process: mean

- Applying the expected value to the definition of AR(1) we get:

$$\mathbb{E}[Y_t] = c + \phi\mathbb{E}[Y_{t-1}] + \mathbb{E}[\varepsilon_t]$$

- Assuming the mean is constant:

$$\mu = c + \phi\mu$$

so:

$$\mu = \frac{c}{1-\phi}$$

AR(1) process: in terms of mean

- Rewriting the AR(1) model:

$$Y_t = c + \phi Y_{t-1} + \varepsilon_t$$

using the relationship for the mean:

$$\mu = \frac{c}{1-\phi}$$

we get:

$$Y_t = \mu(1-\phi) + \phi Y_{t-1} + \varepsilon_t$$

- Rewriting in terms of difference from the mean:

$$Y_t - \mu = \phi \cdot (Y_{t-1} - \mu) + \varepsilon_t$$

- MEM: It is like random walk but with a mean μ and a param ϕ

AR(1) process: properties

- We can compute the statistical properties of AR(1) process using the definition of AR(1) model in terms of the mean:

$$
\begin{aligned}
&\mathbb{E}[Y_t] = \mu \\
&\mathbb{V}[Y_t] = \frac{\sigma_\varepsilon^2}{1-\phi^2} \\
&\mathrm{Cov}[Y_t, Y_{t-j}] = \mathbb{V}[Y_t]\phi^j \\
&\rho(Y_t, Y_{t-j}) = \phi^j
\end{aligned}
$$

- The AR(1) model is weakly stationary if $-1 < \phi < 1$

Ergodicity: intuition

- Y_t and Y_{t-j} tend to being independent as j grows large enough

AR(1) process approximates ergodicity

- The autocorrelation has a geometric decay:

$$\text{Cov}[Y_t, Y_{t-j}] = \mathbb{V}[Y_t]\phi^j$$

  i.e., variables that are closer in time are more correlated than variables that are farther in time

- If $j \to \infty$ then $\text{Cov}[Y_t, Y_{t-j}] \to 0$ (ergodicity)

AR(1) process is mean-reverting

- Mean-reverting = when it is far from the mean, it tends to go back
- The speed of mean reversion depends on ϕ

AR(1) process vs Gaussian white noise

- The AR(1) process is smoother than the GWN due to the autocorrelation in time
- The Gaussian white noise is choppy

AR(1) as function of ϕ

- $\phi = 0 \rightarrow$ white noise: it bounces around the mean
- $0 < \phi < 1$ it stays far from the mean and then reverts (it is smoother)
- $\phi = 1 \rightarrow$ random walk: it walks away from the mean
- $\phi > 1 \rightarrow$ explosive progress since it diverges accelerating
- $\phi < 0$ it is super choppy

AR(1) to model financial time series

- Good model
  - Interest rates
  - Growth rate of macroeconomic variables (growth of GDP, growth of unemployment)
  - Pnl
- Bad model
  - Stocks don't show a strong time dependency
    - Returns look like White noise, prices look like Random walk

AR(p) model

- AR(p) is an autoregressive model of order p:

$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \varepsilon_t$$

  where ε terms are white noise

- MEM: AR(p) models are linear combination of p past realization + noise

AR(p) model in terms of lag operator

- The AR equation is:

$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \varepsilon_t$$

- Separating var and noise term

$$Y_t - \sum \phi_i Y_{t-i} = c + \varepsilon_t$$

- Using lag operator

$$\begin{aligned} Y_t - \sum \phi_i L^i Y_t &= \\ (1 - \sum \phi_i L^i) Y_t &= \\ f(\phi, L) Y_t &= c + \varepsilon_t \end{aligned}$$

Moving average models

- Time Series
- Classical Methods
  - Simple models for stochastic process
  - Autoregressive models
  - **Moving average models**
  - ARMA(p, q) process
  - ARIMA model
  - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

Moving Average (MA) Models

- A MA model of order q predicts future values using past q errors

$$y_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

  where:
  - μ is the mean of the series
  - ϵ_t is the white noise error term at time t
  - $\theta_1, \theta_2, \ldots, \theta_q$ are the parameters of the model
- E.g., correcting for sensor noise by using past error patterns
  - If a sensor consistently overestimates by a small amount, the MA model can adjust for this by considering past errors
- MA models are always stationary
  - Suitable for time series data where the impact of a shock is short-lived
  - Useful for modeling time series with short-term dependencies

MA(1) process: def

- Aka "moving average of order 1"
- MA(1) model is defined as:

$$Y_t = c + \theta\varepsilon_{t-1} + \varepsilon_t$$

  where the noise is iid Gaussian: $\varepsilon_t \sim \text{GWN}(0, \sigma_\varepsilon^2)$
- MEM: MA(1) $=$ linear combination of 1 past innovations $+$ noise
- MEM: MA uses θ like in MAT

MA(1) process: why called moving average?

- Consider:

$$Y_t = c + \phi\varepsilon_{t-1} + \varepsilon_t$$

$$Y_{t-1} = c + \phi\varepsilon_{t-2} + \varepsilon_{t-1}$$

- You can see that it's like a window
  - with given coefficients (computing an average)
  - moving in time

MA(1) process: correlation structure

- There is correlation only between Y_t and Y_{t-1}, but not between any other variable:

$$
\begin{aligned}
& Y_t = f(\varepsilon_t, \varepsilon_{t-1}) \\
& Y_{t-1} = f(\varepsilon_{t-1}, \varepsilon_{t-2}) \\
& Y_{t-2} = f(\varepsilon_{t-2}, \varepsilon_{t-3}) \\
& \dots \\
& Y_{t-k} = f(\varepsilon_{t-k}, \varepsilon_{t-k-1})
\end{aligned}
$$

  since there are common terms only between variables that have a distance $t_1 - t_2 \leq 1$

MA(1) process: properties

- Using the definitions we obtain:

$$\mathbb{E}[Y_t] = c$$
$$\mathbb{V}[Y_t] = (1+\theta)\sigma_\varepsilon^2$$
$$\mathrm{Cov}[Y_t, Y_{t-1}] \stackrel{d.as}{=} \gamma_1 = \theta\sigma_\varepsilon^2$$
$$\mathrm{Cov}[Y_t, Y_{t-j}] \stackrel{d.as}{=} \gamma_j = 0,\ \forall j > 1$$

- It is a weakly stationary process since
  - mean and variance are constant
  - the covariance depends only on the difference of the lags

MA(1) process: example of overlapping returns

- Assume that the 1-month continuously compounded returns r_t are IID normal:

$$r_t \sim \text{ IID N}(\mu_r, \sigma_r^2)$$

- If we consider a monthly time series of 2-month cc returns using:

$$r_t(2) = r_t + r_{t-1}$$

- Then $\{r_t(2)\}$ follows a MA(1) process

MA(q) model

- MA(q) is a moving average model of order q:

$$Y_t = c + \sum_{i=1}^{p} \theta_i \varepsilon_{t-i} + \varepsilon_t$$

  where ε terms are white noise

- Note that c is the mean and so it can be indicated with μ
- It shows autocorrelation among various Y_t terms
- MEM: MA(q) models are linear combination of q error terms from the past

MA(q) model: intuition of covariance structure

- In general MA(q) has dependency between consecutive terms Y_t up to Y_{t-q}
- It can be seen by considering

$$\begin{aligned} Y_t &= f(\varepsilon_t, \varepsilon_{t-1}, ..., \varepsilon_{t-q}) \\ &... \\ Y_{t-k} &= f(\varepsilon_{t-k}, \varepsilon_{t-k-1}, ..., \varepsilon_{t-k-q}) \end{aligned}$$

and noticing that there are common terms as long $t-k \leq t-q \iff k \leq q$

MA(q) model in terms of lag operator

- The MA equation is:

$$Y_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

- Using the lag operator:

$$Y_t = \mu + (1 + \sum_{i=1}^{q} \theta_i L^i)\varepsilon_i = \mu + f(\theta, L)\varepsilon_i$$

ARMA(p, q) process

- Time Series
- Classical Methods
  - Simple models for stochastic process
  - Autoregressive models
  - Moving average models
  - **ARMA(p, q) process**
  - ARIMA model
  - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

ARMA(p, q) model

- It contains p autoregressive terms and q moving average terms:

$$ARMA(p, q) = AR(p) + MA(q)$$

- A realization of an ARMA(p, q) process is:

$$\begin{aligned} Y_t &= AR(p) + MA(q) \\ &= (c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \varepsilon_t) + (c + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t) \\ &= c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t \end{aligned}$$

- AR part involves regressing the variable on its own lagged values
- MA part models error term as a linear combination of lagged error terms

ARMA model in terms of lag operator

- We can separate the terms relative to the variable Y_t and to the error term:

$$(1 - \sum_{i=1}^{q} \phi_i L^i) Y_t = c + (1 + \sum_{i=1}^{p} \theta_i L^i) \varepsilon_i$$

Residuals of ARMA model

- Residuals should be uncorrelated and normally distributed
- One can check the ACF of the residuals

ARMA, ARIMA Models

- ARMA models combine AR and MA components:

$$y_t = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t$$

- ARIMA models extend ARMA by including differencing
  - Handles non-stationary data
  - Useful for time series forecasting
  - Can model a wide range of time series data

ARIMA

- Consider ARIMA(p, d, q) where:
  - $p =$ number of autoregressive terms (AR)
  - $d =$ order of differencing (I)
  - $q =$ number of moving average terms (MA)
- ARIMA(p, d, q) has form:

$$\phi(B)(1 - B)^d y_t = \theta(B)\varepsilon_t$$

  where:
  - $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ is autoregressive (AR) term
  - $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ is moving average (MA) term
    - $B()$ is the backshift operator: $By_t = y_{t-1}$
    - $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$ is white noise
- **Important points:**
  - Differencing helps to stabilize the mean of a time series
  - Over-differencing can lead to increased model complexity without improving accuracy
  - Under-differencing can result in a non-stationary series
  - E.g., if a time series shows a linear trend, first-order differencing ($d = 1$) might be sufficient to achieve stationarity
- Model building steps
  - Identification (select p, d, q)
  - Estimation (fit model)

SARIMA

- Seasonal ARIMA (SARIMA) extends ARIMA to handle seasonal patterns in time series data
- It incorporates seasonal autoregressive and moving average terms, as well as seasonal differencing

ARIMA model

- Time Series
- Classical Methods
  - Simple models for stochastic process
  - Autoregressive models
  - Moving average models
  - ARMA(p, q) process
  - **ARIMA model**
  - ARCH model
- Advanced and Modern Approaches
- Special techniques for time series modeling

ARIMA model class

- = class of statistical models for analyzing and forecasting time series data
- It is a generalization of ARMA (Auto-Regressive Moving Average)
- AR = Auto-Regression
  - uses relationship between next observation and a number of lagged observations
- I = Integrated
  - uses differencing of observations to make the time series stationary
- MA = Moving Average
  - uses the dependency between next observation and a residual error from a moving average model applied to lagged observations

ARIMA(p, d, q)

- p: number of lag observations included in the model
  - aka lag order
- d: degree of differencing (i.e., the number of times the observations are differenced)
- q: size of the moving average window
  - aka order of moving average

Particular cases of ARIMA

- Setting p, d, or q to 0, ARIMA is simplified to a ARMA, AR, I, MA model
- ARIMA(0, 0, 0)
  - $\rightarrow X_t = \varepsilon_t$, which is white noise
- ARIMA(0, 1, 0) $=$ I(1)
  - $X_t = X_{t-1} + \varepsilon_t$, which is a random walk
- ARIMA(p, 0, q) $=$ ARMA(p, q)

ARIMA model in the form of ARMA model

- An ARIMA model can be represented as an ARMA model applied to the time series resulting from differencing
- An ARIMA(p, d, q) is described by the equations:

$$\begin{cases} Z_t = (1-L)^d Y_t \\ (1 - \sum_{i=1}^{p} \phi_i L^i) Z_t = (1 + \sum_{i=1}^{q} \theta_i L^i) \varepsilon_t \end{cases}$$

- So there is differencing of order i, then AR(p) and MA(q)

Fitting ARMA / ARIMA models

- The original Box-Jenkins approach has 3 phases:
  1. Model identification / selection
     - identify seasonality
     - difference data, if necessary, to achieve stationarity
     - check if variables are stationary
     - use ACF, PACF to decide AR and MA components to use
  2. Parameter estimation
     - Pick coefficients to get best fit
  3. Model checking
     - Test estimated model
     - E.g., the residual should have no serial correlation and be stationary in mean and variance
     - If estimation is inadequate, go back to step 1) and attempt to build a better model

ARCH model

- Time Series
- Classical Methods
    - Simple models for stochastic process
    - Autoregressive models
    - Moving average models
    - ARMA(p, q) process
    - ARIMA model
    - **ARCH model**
- Advanced and Modern Approaches
- Special techniques for time series modeling

ARCH: in brief

- $=$ Auto-Regressive Conditional Heteroskedasticity
- ARCH is used to model time series that exhibit time-varying volatility and volatility clustering
- Engle (2003): Nobel price in Economics

Volatility clustering

- $=$ periods of swings interspersed with periods of calm

ARCH: intuition

- Variance of error term (aka innovation) is described as a function of the value of the previous time periods error terms

$$\mathbb{V}[\varepsilon_t] = f(\varepsilon_{t-1}, ..., \varepsilon_{t-N})$$

- E.g., error variance follows an AR model

ARCH(q): definition

- The model for the error term of the time series is:

$$\varepsilon_t = \sigma_t \cdot z_t$$

  where:
  - z_t is white noise process (stochastic part)
  - σ_t^2 is the time-dependent variance given by an AR(q) model:

$$\begin{aligned} \sigma_t^2 &= \alpha_0 + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 \\ &= \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \ldots + \alpha_q \varepsilon_{t-q}^2 \\ &\text{where } \alpha_i \geq 0 \end{aligned}$$

- MEM: the error variance is AR(q), i.e., a linear combination of squares of previous error term realizations

GARCH

- $=$ Generalized ARCH
- The error variance follows an ARMA model

GARCH(p, q): definition

- The error term of a time series is modelled as:

$$\varepsilon_t = \sigma_t \cdot z_t$$

where:
  - z_t is white noise process (stochastic part)
  - σ_t^2 is the time-dependent variance given by an ARMA(p, q) model

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2$$

Seasonal ARIMA (SARIMA)

- SARIMA models capture both non-seasonal and seasonal patterns
- SARIMA notation: ARIMA$(p, d, q)(P, D, Q)_s$
  - (P, D, Q) = seasonal components
  - s = number of periods per season (e.g., $s = 12$ for monthly data)
- Seasonal differencing removes seasonal patterns:
  - $y'_t = y_t - y_{t-s}$
- Useful when strong periodic behavior exists
- Steps similar to ARIMA:
  - Model seasonal and non-seasonal parts separately
- Example: forecasting monthly airline passenger data

Exponential Smoothing Methods

- Forecast future values by weighted averages of past observations
- **Simple Exponential Smoothing**:
  - Good for data with no clear trend or seasonality
- **Holt's Linear Trend Method**:
  - Models both level and trend
- **Holt-Winters Method**:
  - Extends Holt's to include seasonality
- Intuition:
  - More recent observations get more weight
- Forecast equations use smoothing parameters α, β, γ
- Example: predicting daily demand with seasonal shopping patterns

Advanced and Modern Approaches

- Time Series
- Classical Methods
- **Advanced and Modern Approaches**
- Special techniques for time series modeling

State Space Models

- State space models describes how a system evolves over time using states and observations
- **Components**
  - **State vector (x_t):** Hidden/internal state of the system at time t
  - **Observation vector (y_t):** What we can measure at time t
  - **State equation:** $x_{t+1} = F_t x_t + G_t u_t + w_t$
  - **Observation equation:** $y_t = H_t x_t + v_t$
    - F_t: State transition matrix
    - G_t: Control input matrix
    - H_t: Observation matrix
    - w_t, v_t: Process and observation noise
- **Types**
  - **Linear vs Nonlinear**
  - **Time-invariant vs Time-varying**
  - **Deterministic vs Stochastic**
- **Goal**
  - Infer hidden states from noisy observations
  - Predict future observations or states

Vector Autoregressions (VAR)

- **VAR models** generalize AR models to multivariate time series
- Each variable depends on past values of itself and others
- Mathematical form (for 2 variables):
  - $y_{1,t} = c_1 + \phi_{11} y_{1,t-1} + \phi_{12} y_{2,t-1} + \epsilon_{1,t}$
  - $y_{2,t} = c_2 + \phi_{21} y_{1,t-1} + \phi_{22} y_{2,t-1} + \epsilon_{2,t}$
- Intuition:
  - Capture dynamic interrelationships among multiple series
- Used for:
  - Economic indicators
  - Multichannel sensor data
- Example: modeling GDP growth and inflation jointly

Spectral Analysis and Frequency Domain Methods

- Analyze time series in terms of cycles and frequencies
- **Fourier Transform** decomposes series into sinusoidal components
- **Periodogram** estimates strength of different frequencies
- Intuition:
  - Understand repeating patterns that may not be obvious in time domain
- Useful for:
  - Identifying dominant periodicities
  - Filtering noise
- Applications:
  - Seismology, climate cycles
- Example: detect yearly cycle in temperature data

Machine Learning for Time Series

- Use supervised learning to predict future values
- Key steps:
  - Feature engineering (lags, rolling statistics, Fourier terms)
- Common algorithms:
  - Decision trees
  - Random forests
  - Gradient boosting (e.g., XGBoost)
- Handle nonlinearity and complex interactions
- Often requires careful cross-validation due to temporal structure
- Example: predicting electricity consumption using lagged features

Deep Learning for Time Series

- Specialized neural networks for sequential data
- **Recurrent Neural Networks (RNNs)**:
  - Capture dependencies across time steps
- **Long Short-Term Memory networks (LSTMs)**:
  - Solve vanishing gradient problem
  - Retain long-term dependencies
- **Temporal Convolutional Networks (TCNs)**:
  - Use causal convolutions for sequence modeling
- Strengths:
  - Handle complex, nonlinear dynamics
- Require large datasets and careful tuning
- Example: predicting stock price movements using past sequences

Special techniques for time series modeling

- Time Series
- Classical Methods
- Advanced and Modern Approaches
- **Special techniques for time series modeling**

Cross-Validation for Time Series

- Standard cross-validation is **not** suitable due to time dependency
- **Rolling-Origin Evaluation**:
  - Train on expanding window, test on next time step
- **Walk-Forward Validation**:
  - Move training and testing windows forward step-by-step
- Intuition:
  - Always predict the future, never the past
- Allows robust estimation of model performance
- Important for hyperparameter tuning

Anomaly Detection in Time Series

- Identify unusual patterns not consistent with past behavior
- Applications:
    - Finance (fraud detection, unusual trading activity)
    - Cybersecurity (intrusion detection, system failures)
- Common methods:
    - Statistical thresholds (e.g., values $> 3\sigma$ from mean)
    - Machine learning (isolation forests, autoencoders)
- Important to account for seasonality and trend
- Unsupervised methods are often necessary
- Example: detecting a sudden drop in website traffic

Hierarchical and Grouped Time Series Forecasting

- Forecast series that are organized in hierarchies or groups
- **Bottom-up approach**:
  - Forecast each low-level series, aggregate upward
- **Top-down approach**:
  - Forecast top-level series, disaggregate downward
- **Middle-out approach**:
  - Forecast middle levels and adjust both up and down
- Challenges:
  - Coherence (forecasts must add up correctly across levels)
- Applications:
  - Retail (store, region, national sales)
- Example: forecast sales per store, then sum to national level

Probabilistic and Quantile Forecasting

- Predict full distribution of future values, not just a single number
- **Quantile forecasting**:
  - Predict specific quantiles (e.g., 10%, 50%, 90%)
- Helps express uncertainty explicitly
- Useful when risk-sensitive decisions depend on forecast range
- Common methods:
  - Quantile regression
  - Bayesian models
- Example: forecasting a 90% prediction interval for electricity demand