# Notes for Time-Series and Machine Learning Methods
# Part A: Exponential Smoothing and SARIMAX

Last class

Today

Natarajan Gautam

EECS, Syracuse University

# Time-Series Data

- We collect data from observations over time
- We denote the observation numbers (i.e., *index*) as 1, 2, ..., $T$
- What we observe is denoted as $y_1, y_2, \ldots, y_T$
- So $y_t$ for $t \in \{1, \ldots, T\}$ is the $t^{th}$ observation
- For example, the monthly average number of calls, daily average sales per store across locations, amount of solar power generated, etc.
- We wish to forecast $y_{T+h}$ for some pre-defined horizon $h$ which is a positive integer
- That $h$-step ahead forecast for observation $T + h$ given we have observed vales from time 1 to $T$ is represented as $\hat{y}_{T+h|T}$
- Below is a 15-minute average solar irradiance dataset sample

| Index | Date | Time | Avg. Irradiance | |
|---|---|---|---|---|
| 1 | Apr-5-2023 | 11:00-11:14 | 632 | $y_1$ |
| 2 | Apr-5-2023 | 11:15-11:29 | 641 | $y_2$ |
| 3 | Apr-5-2023 | 11:30-11:44 | 579 | $y_3$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $T$ | Apr-12-2023 | 11:15-11:29 | 645 | $y_T$ |

forecast

$\hat{y}_{T+h}$

$y_{T+h} = ?$

# Time-Series Forecast: Exponential Smoothing (ES)

- Why at all exponential smoothing?
  - Handles seasonal data well
  - Unlike SARIMAX, ES very rarely throws errors
  - A special case is indeed persistent forecast
- We begin with the simple time series **without** any trend or seasonality
- A one-step ahead forecast for time $T+1$ given we have observed vales from time 1 to $T$ is represented as $\hat{y}_{T+1|T}$
- The most fundamental exponential smoothing method uses

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \dots$$

  where $\alpha \in [0, 1]$ is called the smoothing parameter
- Note that larger weights are given to most recent observations
- The weights attached to the observations decrease exponentially as we go back in time, hence the name ES

# ES: Component Form for Simple Time Series

- We can show that for any $t$,

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

- The above is written in component form as
  - Forecast equation: $\hat{y}_{t+1|t} = \ell_t$
  - Smoothing equation: $\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$
- We call $y_1$, $y_2$, ..., $y_T$ as the training data
- There are two unknowns $\alpha$ and $\ell_0$
- We select $\alpha$ and $\ell_0$ by minimizing the sum of squared errors (SSE)

$$SSE = \sum_{t=1}^{T} \left(y_t - \hat{y}_{t|t-1}\right)^2$$

- The minimization is usually done with an optimization tool

# ES: Component Form for Time Series with Trend but no Seasonality

- Holt's linear trend method
  - Forecast equation: $\hat{y}_{t+1|t} = \ell_t + b_t$
  - Level equation: $\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$
  - Trend equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$

  $\ell_t$ is an estimate of the level of the series at time $t$, $b_t$ is an estimate of the trend of the series at time $t$, $\alpha$ the smoothing parameter for level, and $\beta^*$ the smoothing parameter for trend
- We could consider other aspects such as *damped* trend, and multiplicative trend (above is additive)
- See Hyndman and Athanasopoulos' FPP3 book for details
- We will not use damping in this course but is an effective tool that prevents extremely high forecast values
- In fact, this case of no seasonality itself is something we do not consider in this course

# ES: Component Form for Time Series with Trend and Seasonality

- Results for any horizon $h$ (earlier we had $h = 1$)
- The seasonality period is $m$ and define $k = \lfloor (h-1)/m \rfloor$
- Holt-Winters' seasonal method uses the iteration
  - Forecast equation: $\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$
  - Level equation: $\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$
  - Trend equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
  - Seasonal equation: $s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$
- Multiplicative seasonality (useful for solar forecast)
  - Forecast equation: $\hat{y}_{t+h|t} = (\ell_t + hb_t)s_{t+h-m(k+1)}$
  - Level equation: $\ell_t = \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}$
  - Trend equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
  - Seasonal equation: $s_t = \gamma\frac{y_t}{\ell_{t-1}+b_{t-1}} + (1-\gamma)s_{t-m}$
- We select $\alpha$, $\beta^*$, $\gamma$, $s_0$, $s_{-1}$ ..., $s_{-m+1}$, $b_0$ and $\ell_0$ by minimizing the RMSE or maximizing the likelihood
- Software also allows for a Box-Cox transformation

# ES: Python Code

- We use `statsmodels.tsa.api` for ES by importing `ExponentialSmoothing`
- Refer to the documentation to see what all parameters can be set
- Check what the default values are for them
- In our case we may have historical data that is much larger than $T$
- We select $T$, train the model, and predict for $T + h$
- Next time we use data from index 2 to $T + 1$, again train the model, and predict for $T + 1 + h$
- We continue this process of sliding window
- We let the model choose $\alpha$, $\beta^*$, $\gamma$, $s_0$, $s_{-1}$ …, $s_{-m+1}$, $b_0$ and $\ell_0$ by the default optimization method
- Technically speaking we may already know some of them as we are considering a sliding window
- We ignore that aspect but note that these can be provided to the function
- For point forecasts we will not do a Box-Cox transformation
- Let us use the code provided in the course

| 2/25 | 2/27 | 3/4 |
| --- | --- | --- |
| Quiz 3 Forecasting | Practice midterm | In-class midterm |

# Time-Series Forecasts: SARIMAX

- We begin by explaining ARIMA (Auto-Regressive Integrated Moving Average), i.e., without seasonality and exogenous data
- The approach taken in ARIMA is to first remove the effects of seasonality and trend using a process called differencing

Main focus

- Then ARIMA simply uses recency to make the forecasts, and then re-introduce the seasonality and trend
- Hence, as an initial step, we try to create a *stationary* time-series
- A stationary time-series in an ideal condition is independent of the time stamp (index)
- Both trend and seasonality would render a time-series to be non-stationary and they would have to be removed
- ARIMA assumes the time-series to be stationary
- The approach used to create a "nearly" stationary time-series is called differencing

## Differencing

- Recall that we have observations $y_1$, $y_2$, ..., $y_T$ that we use for making the forecasts
- We denote the time series as $\{y_t : 1 \leq t \leq T\}$
- Now consider the time-series $\{y'_t : 2 \leq t \leq T\}$ which is called the first-order differencing term defined as

$$y'_t = y_t - y_{t-1}$$

1st differencing

for all $t \in \{2, 3, \ldots, T\}$

- The differenced series has one fewer term than the original time-series
- Sometimes the series $\{y'_t : 1 \leq t \leq T\}$ may not be adequately stationary, and it may be required to do one more round of differencing
- For that, consider the time series $\{y''_t : 2 \leq t \leq T\}$ defined as

$$y''_t = y'_t - y'_{t-1}$$

2nd differencing

for all $t \in \{3, 4, \ldots, T\}$

- Note that the second order differenced series has one fewer term than the first-order ~~time series~~ differencing and two fewer than the original time series

# Differencing Example

## Differencing: Some Comments and Seasonal Differencing

- In the previous figure, the automated code using `pmdarima` picks first order
- Usually first order differencing is enough, rarely second order
- It is not recommended to go beyond second order differencing, although the series may not be stationary
- Sometimes that is because of lingering seasonal effects
- When there are seasonal effects, it may be worthwhile (although not guaranteed to remove) to perform seasonal differencing
- Let $\{y_t^s : m \leq t \leq T\}$ be lag-$m$ differences, where $m$ is the period of seasonality
- The seasonal differencing term is defined as

$$y_t^s = y_t - y_{t-m}$$

  for all $t \in \{m+1, m+2, \ldots, T\}$
- If needed, we could perform a second-differences (one against season and one against consecutive terms)

# Time-Series Forecasts: Model for ARIMA

- We begin with ARIMA with no seasonal effects.
- We stationarize the time series using differencing

$d = 0$ or $1$ or $2$

- The **ARIMA($p, d, q$)** model has $p$ as the autoregressive part, $d$ the degree of differencing and $q$ the moving average part
- This gives rise to the model (in terms of the differenced series $y'_t$)

$$y'_t = c + \phi_1 y'_{t-1} + \ldots + \phi_p y'_{t-p} + \underline{\underline{\theta_1 \epsilon_{t-1}}} + \ldots + \underline{\underline{\theta_q \epsilon_{t-q}}} + \underline{\underline{\epsilon_t}}$$

where $\epsilon_t$ is from the original series

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

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

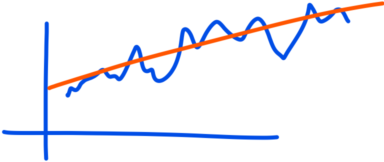
- Also, the $\epsilon_t$ terms are a function of $c$, $\phi_1$, ..., $\phi_p$ as well as the time series
- Note that $c$, $\phi_1$, ..., $\phi_p$, $\theta_1$, ..., $\theta_q$ are all unknown

# Time-Series Forecasts: Estimating the ARIMA Unknowns

Assume we have selected $p$ and $q$ (and $d$)

- In the ARIMA model we can use MLE (default in the python library we use) to estimate $c$, $\phi_1$, ..., $\phi_p$, $\theta_1$, ..., $\theta_q$ which are all unknown
- For the MLE, we maximize the likelihood of obtaining the series $y'_t$ for all $t$ assuming it is Gaussian and so are the $\epsilon_t$ values
- Another approach is to minimize the sum of squares of $\epsilon_t$
- Either case would result in a complicated non-linear optimization problem that cannot be written in closed-form
- There are numerous software packages that produce the results

## Time-Series Forecasts: Model for SARIMAX



- The key idea of seasonal-ARIMA is to add seasonal terms
- Extend the **ARIMA($p, d, q$)** model to the **seasonal-ARIMA**$\underbrace{(p, d, q)}_{\text{ARIMA}}\underbrace{(P, D, Q)}_{\text{seasonal}}$ with $m$ seasons
- Although we do not present the equation, it is worthwile understanding that we add seasonality AR terms $y_{t-m}$, .... $y_{t-Pm}$ with coefficients $\Phi_1$, ..., $\Phi_P$, as well as seasonality MA terms $\epsilon_{t-m}$, ..., $\epsilon_{t-Qm}$ with coefficients $\Theta_1$, ..., $\Theta_Q$
- Now, if we were to add exogenous variables, say $x_1$, $x_2$, ..., $x_r$, then we essentially get the form

$$y'_t = c + b_1 x_1 + b_2 x_2 + \ldots + b_r x_r + \phi_1 y'_{t-1} + \ldots$$

# Selecting Model Parameters for SARIMAX

$d, D$ by using pmdarima or simply plotting.

- Next we need to determine how to get $p$, $q$, $P$, and $Q$
- They can be selected by searching through the space of values and find the one that minimizes an information cretrion
- The common choice of information criteria are AIC (Akaike's Information Criterion), AICc (i.e. AIC corrected), and BIC (Bayesian Information Criterion), see Hyndman and Athanasopoulos' FPP3 book
- We do not present the expressions here but they are typically functions of likelihood of obtain the data (that was used in MLE), $T$, $p$, $q$, $P$, $Q$, and whether or not $c$ was used
- There is also an automated way (although many do not recommend) to do this using `pmdarima`

# SARIMAX Forecasts: Python Code

- Import various python tools and libraries
- Select params viz. test data, sample size, etc.
- Get data and aggregate, if necessary
- Clean up data by imputing
- Merge with exogenous data (if we use them) to get required dataset
- Visualize the data for a few days
- Fit auto-arima model, or other methods for $(p, d, q)$ and $(P, D, Q)$
- Use sliding window for predictions
- Test predictions using various metrics