01Introduction

February 11, 2022

```
[1]: import warnings
     warnings.filterwarnings('ignore')
```

QBUS3850: Time Series and Forecasting

Introduction

Lecture Notes

Housekeeping

1 Teaching Team

- Lectures (Monday 10-12)
  - Lecturer: Anastasios Panagiotelis
- Tutorials (Tuesday 10-12, 2-4)
  - Tutor: Tin Lai
- Contact details and consultation times can be found on Canvas
- Lectures will be recorded, tutorials will not be

2 Textbooks

- We will stick fairly close to course materials but the following books can *supplement* your learning
  - Hyndman, Rob J and George Athanasopoulos, *Forecasting: Principles and Practice*. Free online
  - Tsay, Ruey, (2010), *Analysis of Financial Time Series*, Available from major retailers.

3 Software Tools

- In this course we use Python
- Lecture slides have embedded Python code and can be used interactively
- Tutorials are built on Jupyter Notebooks
- The emphasis is on forecasting in *practice* as well as the theory of forecasting

4 Assessment

- Three components:

  - Mid-semester exam (20%)
  - Major (group) assignment (40%)
  - Final exam (40%)
- More details on Canvas

Let's Get Started!

5 Time Series and Forecasting

- The focus of this course is **forecasting** (what a suprise).
- In this course we mostly consider numerical variables
  - Example: electricity demand
- Will mostly use the history of the variable we want to forecast or **time series** itself.
  - Make a forecast $\hat{y}_{T+h}$ targeting y_{T+h}
  - Use $y_1, y_2, \ldots, y_T$
- Here h is the forecast horizon.

6 Forecasting v Prediction

- In other courses you have made predictions $\hat{y} = f(\mathbf{x})$, where $\mathbf{x}$ are some independent variables.
- A forecast is just a prediction in the future
- Any method can be used if $f(\mathbf{x})$ includes information available when the forecast is made
- Unlike general regression methods, in time series forecasting there are techniques to exploit the temporal structure or patterns in the data.

7 What can we easily forecast?

In general it is easier to forecast a variable when:

1. We understand the factors that contribute to it;
2. When more data are available;
3. When the future is similar to the past;
4. When the forecasts do not affect the quantity we are trying to forecast.

8 Your turn

- In small groups have a think about:
  - Variables that we need to forecast,
  - How forecasting these variables can improve business decisions,
  - Whether it is easy to forecast these variable.
- Discuss in groups for 10 minutes and then we will discuss together.

An example (with a review of exponential smoothing)

9 Electricity Demand

- Forecasts at different **horizons** play an important role in the operation of electricity markets.

- Long term forecasts (10 to 30 years horizon)
  - Used by by policy makers or investors to decide how to invest in infrastructure.
- Short term forecasts (up to 24-hour horizon)
  - Used for operations, e.g. scheduling which generators produce electricity.
- Consider the short term forecasting problem to revise exponential smoothing models covered last semester.
- Data sourced from the Australian Energy Market Operator (AEMO) and on Canvas.

10 Electricity Data

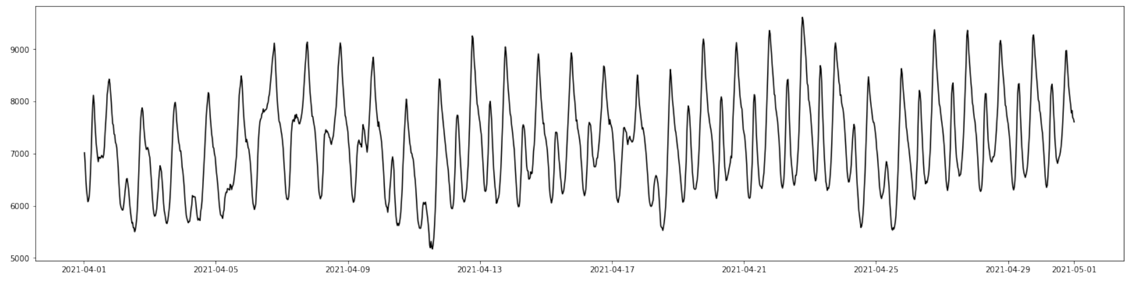
[2]:
```
import pandas as pd
elec = pd.read_csv('electricity.csv', header=0, index_col=False,␣
 ↪parse_dates=False)
elec['SETTLEMENTDATE'] = pd.to_datetime(elec['SETTLEMENTDATE'])
elec = elec[['SETTLEMENTDATE','TOTALDEMAND']]
print(elec.head())
```

```
       SETTLEMENTDATE  TOTALDEMAND
0 2021-04-01 00:30:00      7012.00
1 2021-04-01 01:00:00      6815.37
2 2021-04-01 01:30:00      6495.25
3 2021-04-01 02:00:00      6308.43
4 2021-04-01 02:30:00      6179.93
```

11 Line plot (1 month)

[3]:
```
import matplotlib.pyplot as plt
fig, ax = plt.subplots(figsize=(25, 6))
ax.plot(elec.SETTLEMENTDATE, elec.TOTALDEMAND, color='black')
```

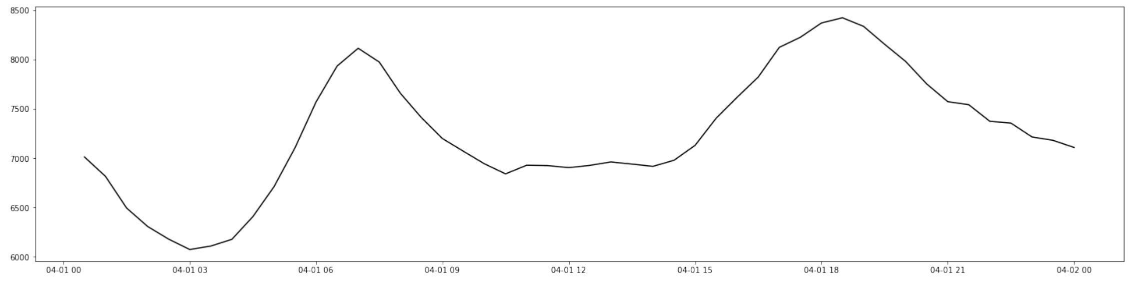
[3]:
```
[<matplotlib.lines.Line2D at 0x7f421e7ef670>]
```



12 Line plot (1 day)

[4]:
```
elec_ow = elec.head(n=2*24)
fig, ax = plt.subplots(figsize=(25, 6))
ax.plot(elec_ow.SETTLEMENTDATE, elec_ow.TOTALDEMAND, color='black')
```

[4]:
```
[<matplotlib.lines.Line2D at 0x7f421dab8a30>]
```



13 Patterns

- Electricity demand is lower:
  - Overnight/ middle of day,
  - On weekends and public holidays,
  - In months with moderate temperature (not shown in plots above)
- All this can be deduced just by looking at the series itself!

14 Simple Exponential Smoothing

- Simple (no trend, no seasonality)

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

- Where $0 \le \alpha \le 1$ and l_0 need to be estimated.
- The logic: Forecast is an average level of the time series where more recent observations are given more influence in determining that level.

14.0.1 Data Cleaning

The following code creates a test sample of the final day (48 half hours) and the remaining data is used for training.

[5]:
```
import numpy as np
from statsmodels.tsa.holtwinters import SimpleExpSmoothing, Holt,␣
 ↪ExponentialSmoothing
elec_clean = elec.set_index('SETTLEMENTDATE')
```

```
train = elec_clean.iloc[0:-48, :]
test = elec_clean.iloc[-48:, :]
```

14.0.2 Fit model (Simple ES)

The following code fits the model

[6]:
```
model = SimpleExpSmoothing(np.asarray(train['TOTALDEMAND']))
fit = model.fit()
fit.summary()
```

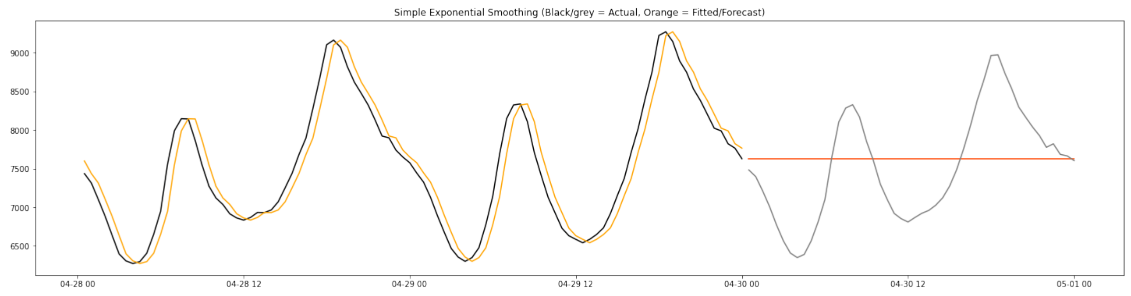
```
/home/anastasios/anaconda3/lib/python3.8/site-
packages/statsmodels/tsa/holtwinters/model.py:920: ConvergenceWarning:
Optimization failed to converge. Check mle_retvals.
  warnings.warn(
```

[6]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                       SimpleExpSmoothing Model Results
==============================================================================
Dep. Variable:                    endog   No. Observations:                 1392
Model:               SimpleExpSmoothing   SSE                       57739558.709
Optimized:                         True   AIC                          14805.075
Trend:                             None   BIC                          14815.552
Seasonal:                          None   AICC                         14805.104
Seasonal Periods:                  None   Date:                 Fri, 11 Feb 2022
Box-Cox:                          False   Time:                         10:26:50
Box-Cox Coeff.:                    None
==============================================================================
                       coeff                 code              optimized
------------------------------------------------------------------------------
smoothing_level            0.9950000                alpha                 True
initial_level              7012.0000                  1.0                 True
------------------------------------------------------------------------------
"""
```

14.0.3 Plot (Simple ES)

[7]:
```
pred = test.copy()
pred = fit.forecast(48)
fitted = fit.fittedvalues
fig, ax = plt.subplots(figsize=(25, 6))
ax.plot(train.index[-96:], train.values[-96:],color='black')
ax.plot(train.index[-96:], fitted[-96:], color='orange')
ax.plot(test.index, pred,color='orangered')
ax.plot(test.index, test.values, color='grey')
```

```
plt.title("Simple Exponential Smoothing (Black/grey = Actual, Orange = Fitted/
↪Forecast)");
```



15 Holt exponential smoothing

- Corrects for trend

$$\begin{aligned}\hat{y}_{t+h} &= l_t + hb_t \\ l_t &= \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}\end{aligned}$$

- where $0 \le \alpha \le 1$, $0 \le \beta \le 1$ and l_0 and b_0 need to be estimated.
- The logic: There is a trend also varying smoothly over time. Instead of flat forecasts, forecasts should be linear. The “local” trend is just the difference between the two most recent observations.

15.0.1 Fit model (Holt)

The following code fits the model

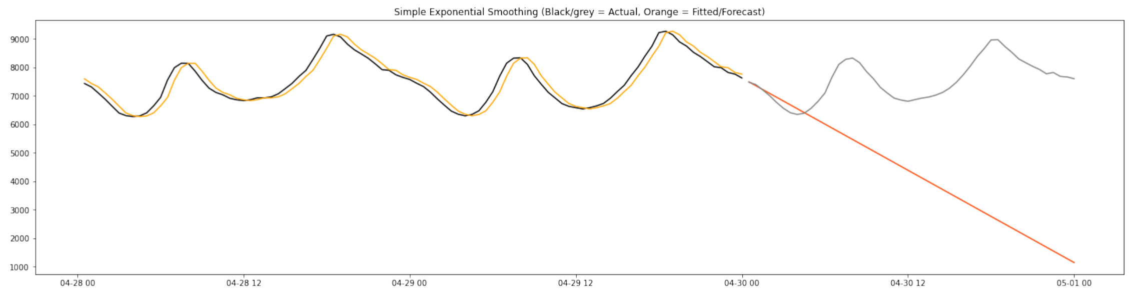
```
[8]: model = Holt(np.asarray(train['TOTALDEMAND']))
     fit = model.fit()
     fit.summary()
```

```
[8]: <class 'statsmodels.iolib.summary.Summary'>
     """
                                Holt Model Results
     ==============================================================================
     Dep. Variable:                    endog   No. Observations:                 1392
     Model:                             Holt   SSE                      17201041.078
     Optimized:                         True   AIC                         13123.401
     Trend:                         Additive   BIC                         13144.355
     Seasonal:                          None   AICC                        13123.462
     Seasonal Periods:                  None   Date:                 Fri, 11 Feb 2022
     Box-Cox:                          False   Time:                         10:26:50
     Box-Cox Coeff.:                    None
```

```
================================================================================
                        coeff                code              optimized
--------------------------------------------------------------------------------
smoothing_level            0.9950000                alpha                True
smoothing_trend            0.9950000                 beta                True
initial_level              7012.0000                  l.0                True
initial_trend             -196.63000                  b.0                True
--------------------------------------------------------------------------------
"""
```

15.0.2 Plot (Holt)

```
[9]: pred = fit.forecast(48)
     fig, ax = plt.subplots(figsize=(25, 6))
     ax.plot(train.index[-96:], train.values[-96:],color='black')
     ax.plot(train.index[-96:], fitted[-96:], color='orange')
     ax.plot(test.index, pred,color='orangered')
     ax.plot(test.index, test.values, color='grey')
     plt.title("Simple Exponential Smoothing (Black/grey = Actual, Orange = Fitted/
      ↪Forecast)");
```



16 Holt-Winters exponential smoothing

- Corrects for trend and seasonality. Additive form is:

$$
\begin{aligned}
\hat{y}_{t+h} &= l_t + hb_t + s_{t+h-m(k+1)} \\
l_t &= \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1}) \\
b_t &= \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1} \\
s_t &= \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}
\end{aligned}
$$

- where $0 \leq \alpha \leq 1$, $0 \leq \beta \leq 1$, $0 \leq \gamma \leq 1$ and l_0 and b_0 and $s_0, s_{-1}, s_{-2}, ...$ need to be estimated.
- Note m is the seasonal period and k is the integer part of $(h-1)/m$.

17 Seasonal period

- Think of *seasonal period* as the number of observations required for a pattern to repeat itself
- For example monthly data has a seasonal period of 12, for a pattern repeating every year
- In our example the period repeats every day.
- With half hourly data that implies a seasonal period of 48.
- An argument could also be made for a seasonal period of $48 \times 7 = 336$ or $48 \times 365 = 17520$.

17.0.1 Fit model (Holt-Winters)

The following code fits the model

[10]:
```
model = ExponentialSmoothing(np.
 ↪asarray(train['TOTALDEMAND']),trend='add',seasonal='add',seasonal_periods=48)
fit = model.fit()
fit.summary()
```

[10]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                    ExponentialSmoothing Model Results
================================================================================
Dep. Variable:                    endog   No. Observations:                 1392
Model:             ExponentialSmoothing   SSE                        9027235.473
Optimized:                         True   AIC                          12321.946
Trend:                         Additive   BIC                          12594.348
Seasonal:                      Additive   AICC                         12326.389
Seasonal Periods:                    48   Date:                 Fri, 11 Feb 2022
Box-Cox:                          False   Time:                         10:26:51
Box-Cox Coeff.:                    None
================================================================================
=
                          coeff                 code              optimized
--------------------------------------------------------------------------------
-
smoothing_level           0.9268520                alpha
True
smoothing_trend           0.8947534                 beta
True
smoothing_seasonal        0.0437029                gamma
True
initial_level             7175.3224                  l.0
True
initial_trend            -7.7438693                  b.0
True
initial_seasons.0        -232.50315                  s.0
True
initial_seasons.1        -378.25651                  s.1
True
```

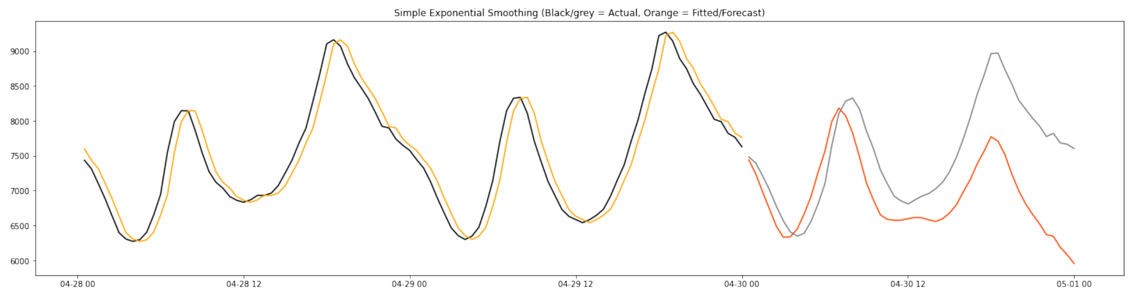
```
initial_seasons.2           -601.84968                  s.2
True
initial_seasons.3           -796.55259                  s.3
True
initial_seasons.4           -994.37038                  s.4
True
initial_seasons.5           -1075.7827                  s.5
True
initial_seasons.6           -1019.0359                  s.6
True
initial_seasons.7           -886.66038                  s.7
True
initial_seasons.8           -676.34155                  s.8
True
initial_seasons.9           -435.62796                  s.9
True
initial_seasons.10          -102.11396                 s.10
True
initial_seasons.11           241.22203                 s.11
True
initial_seasons.12           635.36301                 s.12
True
initial_seasons.13           832.00894                 s.13
True
initial_seasons.14           805.52243                 s.14
True
initial_seasons.15           639.95241                 s.15
True
initial_seasons.16           394.14167                 s.16
True
initial_seasons.17           124.47445                 s.17
True
initial_seasons.18          -65.120165                 s.18
True
initial_seasons.19          -222.96612                 s.19
True
initial_seasons.20          -272.42713                 s.20
True
initial_seasons.21          -307.10447                 s.21
True
initial_seasons.22          -312.37807                 s.22
True
initial_seasons.23          -279.83256                 s.23
True
initial_seasons.24          -249.60984                 s.24
True
initial_seasons.25          -239.01125                 s.25
```

```
True
initial_seasons.26          -219.68854                s.26
True
initial_seasons.27          -176.79816                s.27
True
initial_seasons.28          -89.543995                s.28
True
initial_seasons.29           9.6078275                s.29
True
initial_seasons.30           174.54784                s.30
True
initial_seasons.31           397.44353                s.31
True
initial_seasons.32           605.04641                s.32
True
initial_seasons.33           881.61648                s.33
True
initial_seasons.34           1108.4864                s.34
True
initial_seasons.35           1298.0437                s.35
True
initial_seasons.36           1353.8133                s.36
True
initial_seasons.37           1214.1638                s.37
True
initial_seasons.38           947.82317                s.38
True
initial_seasons.39           743.68373                s.39
True
initial_seasons.40           597.28310                s.40
True
initial_seasons.41           451.91726                s.41
True
initial_seasons.42           337.91015                s.42
True
initial_seasons.43           210.44292                s.43
True
initial_seasons.44           211.94110                s.44
True
initial_seasons.45           104.60348                s.45
True
initial_seasons.46           15.200615                s.46
True
initial_seasons.47          -78.151432                s.47
True
--------------------------------------------------------------------------------
-
```

"""

17.0.2 Plot (Holt-Winters)

```
[ ]: pred = fit.forecast(48)
     fig, ax = plt.subplots(figsize=(25, 6))
     ax.plot(train.index[-96:], train.values[-96:],color='black')
     ax.plot(train.index[-96:], fitted[-96:], color='orange')
     ax.plot(test.index, pred,color='orangered')
     ax.plot(test.index, test.values, color='grey')
     plt.title("Simple Exponential Smoothing (Black/grey = Actual, Orange = Fitted/
      ↪Forecast)");
```



Forecast Evaluation

18 Evaluating Forecasts

- In the previous example we had test and training data.
- We fit the model once and made 1-step ahead to 48 step-ahead forecasts.
- We could evaluate the accuracy of these forecasts using mean square error, mean absolute error or other metrics.
- Is this the end of the story?
- Since methods can have very different performance 1-step ahead compared to 48-steps ahead, does it make sense to average across these horizons?

19 Your turn

- Consider a case where only 1-step ahead forecasting is needed.
- Would the above evaluation be the best strategy?
- If not, what would be a better strategy (*Hint: think of leave one out validation*)?
- Discuss for 5 minutes (and don't cheat by looking at the next slide).

20 Rolling or Window

- A common way to evaluate forecasts is via a rolling window.

1. Fit the model from $y_1, ..., y_{T_{\text{train}}}$ and make forecast $y_{T_{\text{train}}+1}$
2. Fit the model from $y_2, ..., y_{T_{\text{train}}+1}$ and make forecast $y_{T_{\text{train}}+2}$
3. Fit the model from $y_3, ..., y_{T_{\text{train}}+2}$ and make forecast $y_{T_{\text{train}}+3}$
4. and so on…

- This gives a series of one-step ahead forecasts to evaluate one-step ahead forecast accuracy.
- We can compute mean square error or mean absolute error over the rolling window.

21 Expanding Window

- A alternative way to evaluate forecasts is via a expanding window.
  1. Fit the model from $y_1, ..., y_{T_{\text{train}}}$ and make forecast $\hat{y}_{T_{\text{train}}+1}$
  2. Fit the model from $y_1, ..., y_{T_{\text{train}}+1}$ and make forecast $y_{T_{\text{train}}+2}$
  3. Fit the model from $y_1, ..., y_{T_{\text{train}}+2}$ and make forecast $\hat{y}_{T_{\text{train}}+3}$
  4. and so on…
- Which do you think is better and why?

22 Rolling v Expanding

- Expanding window mimics the process of using all available data whenever forecasts are made.
  - This is common in practice.
- Rolling windows use the same amount of training data
  - Rolling windows make sense if model parameters are unstable over time.
- Common Sense: *Evaluate forecasts in the way you anticipate you will be using forecasts!*
- Also not, although the expanding/rolling windows described here use one-step ahead forecasts, we can also use multistep ahead forecasts too.

Point v Probabilistic Forecasts

23 Think probabilistically

- We often think of forecasts as single number or *point forecasts*, for example
  - Electricity demand at 3PM tomorrow will be 8000 MWh
- Sometimes decisions may depend of the variance of a forecast, i.e.

$$Var(y_{T+h}|\mathcal{F}_T)$$

- Other decisions may depend on quantiles, i.e.

$$q : \Pr(y_{T+h} < q|\mathcal{F}_T) = \alpha$$

24 Example 1 : Newsvendor problem

- Retailer purchases goods for c dollars, sells them for p dollars, demand is D, quanity ordered is q
- Profit is $\pi = pmin(q, D) - cq$
- If D is normal with mean μ and variance σ^2 the optimal order quantity

$$q_{opt} = \mu + \sigma\Phi^{-1}\left(\frac{p-c}{p}\right)$$

- The optimal decision requires the forecast variance!

25 Example 2 : Value at Risk

- The Basel II accords are a set of international recommendations for the regulation of banks.
- Under this accord, the value at risk (a forecast quantile) is the preferred measure of market risk.
- This is used to determine reserves of capital that should be held.
- Although this requirement has been superseeded by Basel III, value at risk is still common in practice.

26 Example 3 : Everyday Examples

- Bank of England Fan charts (source)
- Rain probability in weather apps.

27 Confidence v Prediction Interval

- A *confidence interval* describes the uncertainty around my forecast due to parameter uncertainty
  - e.g. in a regression model uncertainty due to β
- A *prediction interval* also accounts for the uncertainty since the future outcome is unknown
  - e.g. in a regression model it will reflect the uncetainty in ϵ
- Prediction intervals are wider than confidence intervals

28 Probabilistic Forecasts

- A probabilistic forecast is a density (or probability mass function)

$$f(y_{T+h}|\mathcal{F}_T)$$

- From this we can get forecast variances, quantiles, prediction intervals
- We can also get point forecasts either as
  - The expected value $E(y_{T+h}|\mathcal{F}_T)$
  - The median $q:(y_{T+h} < q|\mathcal{F}_T) = 0.5$

Wrap up

29 Conclusion

- We will learn about many models and methods but certain things always apply
  - Understand the business context when you make forecasts
  - Evaluate your forecasts in a way that reflects how you will use them

– Always give an idea about the uncertainty around forecasts