classical
TS-forecasting

Linear-Regr

Prob: MLE, MAP ✓

Geom & optimization
(ML)

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:13

Forecasting at Uber: An In | Time-Series.ipynb - Colab | Stationary process - Wiki | pandas.DataFrame.rolling | Partial autocorrelation fun | Exponential smoothing - W | Prophet | Forecasting at s | Forecasting at scale [Peer

colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=8o6WbdY1HL_t

+ Code + Text

RAM
Disk

Data

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = (20, 6)
```

```
[2] !gdown 1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
```

```
Downloading...
From: https://drive.google.com/uc?id=1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
To: /content/mobilesales.xlsx
100% 13.7k/13.7k [00:00<00:00, 13.2MB/s]
```

```python
mobile_sales = pd.read_excel('/content/mobilesales.xlsx')
mobile_sales.head()

# Monthly sales data
```

	DATE	Sales
0	2001-01-01	6519.0
1	2001-02-01	6654.0
2	2001-03-01	7332.0

2 / 2

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Forecasting at Uber: An In | Time-Series.ipynb - Colab | Stationary process - Wiki | pandas.DataFrame.rolling | Partial autocorrelation fun | Exponential smoothing - W | Prophet | Forecasting at s | Forecasting at scale [Peer

colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=8o6WbdY1HL_t

+ Code + Text

RAM
Disk

Data

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = (20, 6)
```

[2]
```
!gdown 1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
```

```
Downloading...
From: https://drive.google.com/uc?id=1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
To: /content/mobilesales.xlsx
100% 13.7k/13.7k [00:00<00:00, 13.2MB/s]
```

```python
mobile_sales = pd.read_excel('/content/mobilesales.xlsx')
mobile_sales.head()

# Monthly sales data
```

	DATE	Sales
0	2001-01-01	6519.0
1	2001-02-01	6654.0
2	2001-03-01	7332.0

3 / 3

```
plt.rcParams['figure.figsize'] = (20, 6)
```

```
[2] !gdown 1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
```

```
Downloading...
From: https://drive.google.com/uc?id=1-pOuGRd8zuAUKBll-1xkr7_867NwoWHg
To: /content/mobilesales.xlsx
100% 13.7k/13.7k [00:00<00:00, 13.2MB/s]
```

```
[3] mobile_sales = pd.read_excel('/content/mobilesales.xlsx')
    mobile_sales.head()

    # Monthly sales data
```

monthly sales

	DATE	Sales
0	2001-01-01	6519.0
1	2001-02-01	6654.0
2	2001-03-01	7332.0
3	2001-04-01	7332.0
4	2001-05-01	8240.0

```
[4] mobile_sales.dtypes
```

```
DATE      datetime64[ns]
Sales            float64
dtype: object
```

```
100% 13.7k/13.7k [00:00<00:00, 13.2MB/s]
```

[3]
```
mobile_sales = pd.read_excel('/content/mobilesales.xlsx')
mobile_sales.head()

# Monthly sales data
```

	DATE	Sales
0	2001-01-01	6519.0
1	2001-02-01	6654.0
2	2001-03-01	7332.0
3	2001-04-01	7332.0
4	2001-05-01	8240.0

[4]
```
mobile_sales.dtypes
```

```
DATE      datetime64[ns]
Sales            float64
dtype: object
```

[5]
```
mobile_sales.Sales.describe()
```

```
count      198.000000
mean     10842.757576
std       2605.006293
```

```
DATE        datetime64[ns]
Sales              float64
dtype: object
```

```
[5] mobile_sales.Sales.describe()
```

```
count      198.000000
mean     10842.757576
std       2605.006293
min        108.000000
25%       8856.000000
50%      10745.000000
75%      12399.750000
max      23740.000000
Name: Sales, dtype: float64
```

missing-values

```
[6] mobile_sales.DATE.describe()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Treating datetime data as categorical rather than numeric i
  """Entry point for launching an IPython kernel.
count                     217
unique                    217
top       2001-01-01 00:00:00
freq                        1
first     2001-01-01 00:00:00
last      2019-01-01 00:00:00
Name: DATE, dtype: object
```

```
[7] mobile_sales.index
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Forecasting at Uber: An In | Time-Series.ipynb - Colab | Stationary process - Wiki | pandas.DataFrame.rolling | Partial autocorrelation fun | Exponential smoothing - W | Prophet | Forecasting at s | Forecasting at scale [Peer

colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=VTTEI5osHrug

+ Code + Text

RAM Disk

[6]

```
  """Entry point for launching an IPython kernel.
count                    217
unique                   217
top       2001-01-01 00:00:00
freq                       1
first     2001-01-01 00:00:00
last      2019-01-01 00:00:00
Name: DATE, dtype: object
```

[7]

```
mobile_sales.index
```

```
RangeIndex(start=0, stop=217, step=1)
```

```
mobile_sales.set_index('DATE', inplace=True)
mobile_sales
```

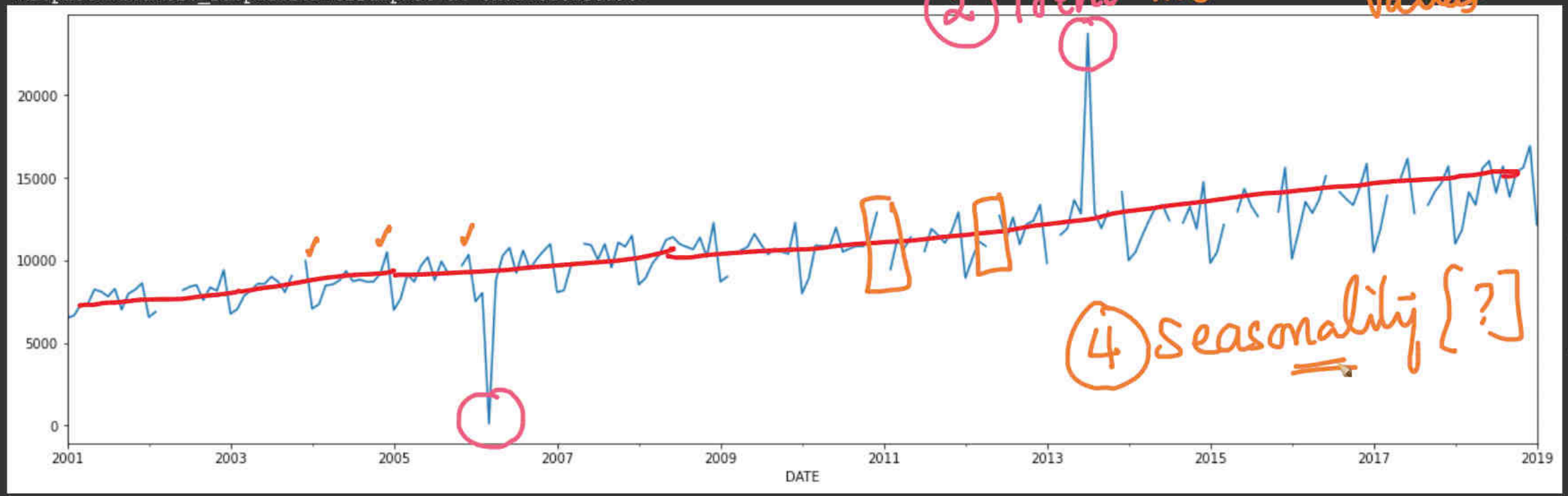
	Sales
DATE	
2001-01-01	6519.0
2001-02-01	6654.0
2001-03-01	7332.0
2001-04-01	7332.0
2001-05-01	8240.0
...	...

7/7

217 rows × 1 columns

```
[9] mobile_sales.Sales.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0b0f50a90>
```



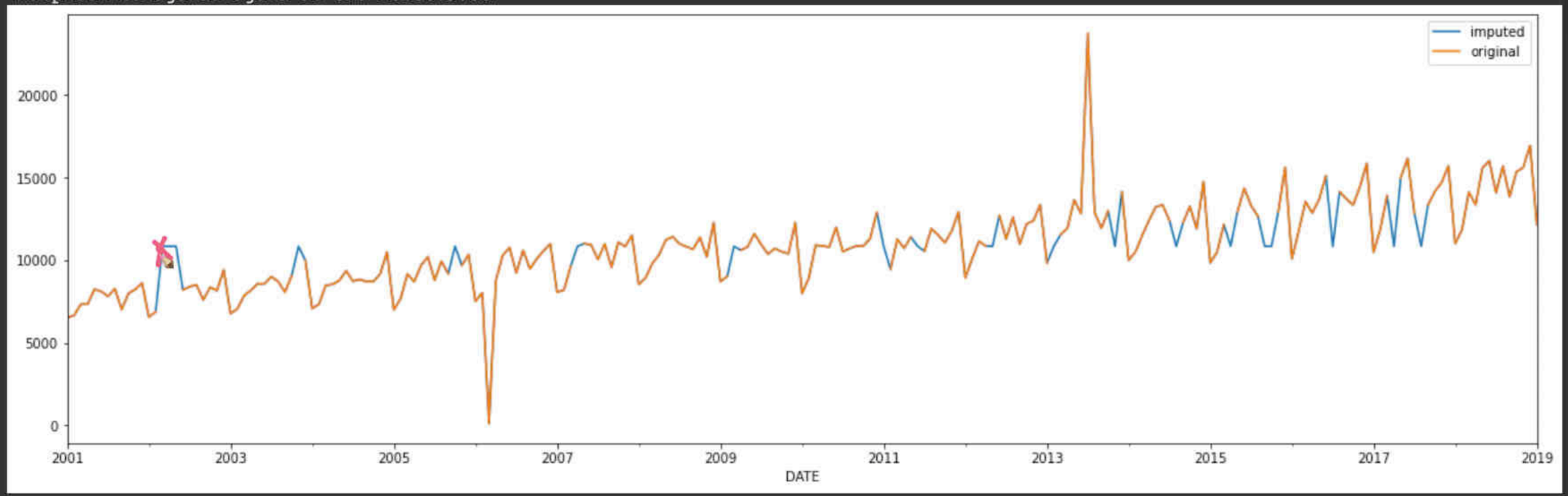
Missing values & Anomalies

[ ] ↳ 11 cells hidden

(Global)
mean-imputation [cons]

```
#mean imputation: what are the issues?
mobile_sales.Sales.fillna(mobile_sales.Sales.mean()).plot(label='imputed')
mobile_sales.Sales.plot(label='original')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ff0b0c49750>
```



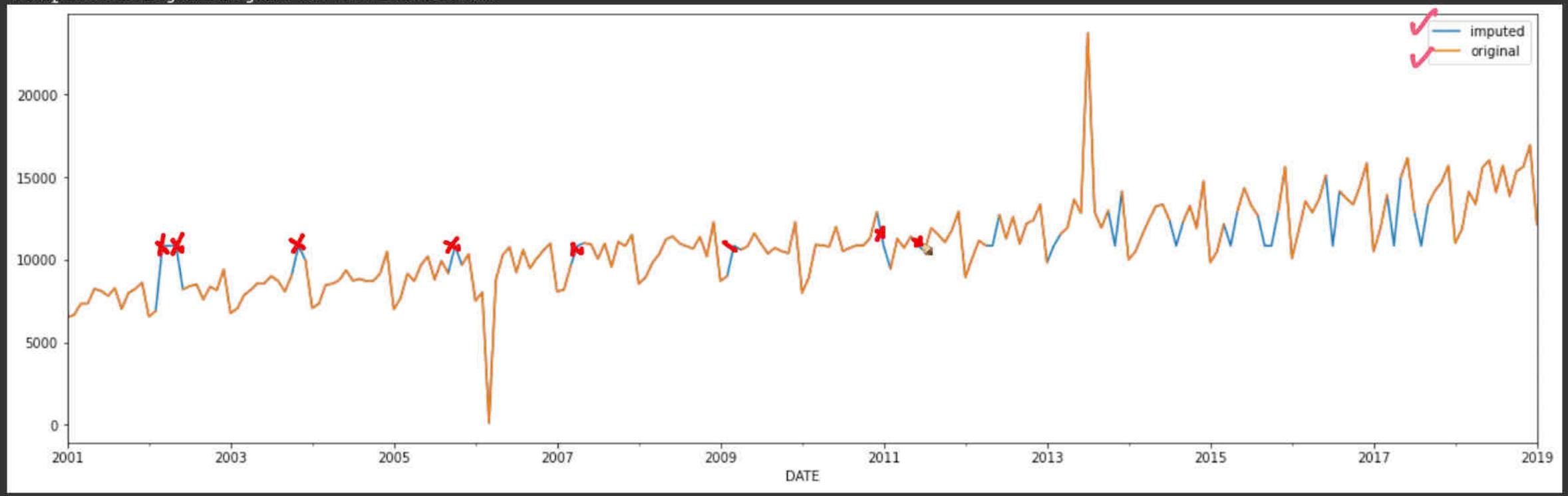

```
[13] # linear interpolation
     mobile_sales.Sales.interpolate(meth
```

[11]

2017-08-01	NaN

[12]
```
#mean imputation: what are the issues?
mobile_sales.Sales.fillna(mobile_sales.Sales.mean()).plot(label='imputed')
mobile_sales.Sales.plot(label='original')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ff0b0c49750>
```

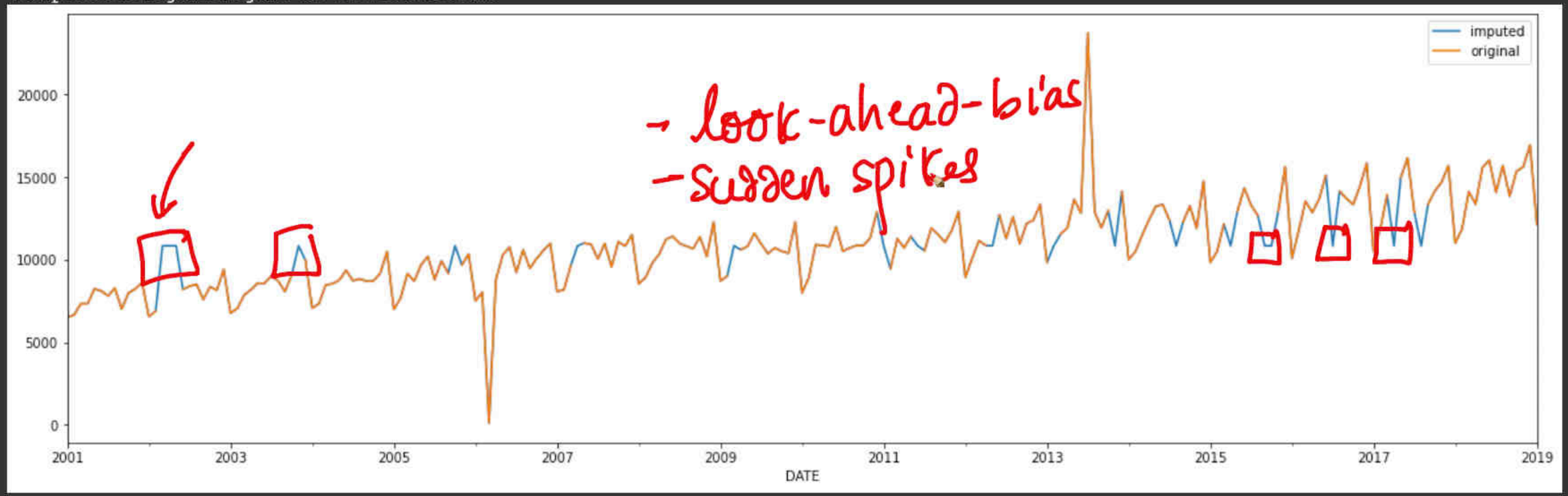


[11]

2017-08-01	NaN

[12]
```
#mean imputation: what are the issues?
mobile_sales.Sales.fillna(mobile_sales.Sales.mean()).plot(label='imputed')
mobile_sales.Sales.plot(label='original')
plt.legend()
```

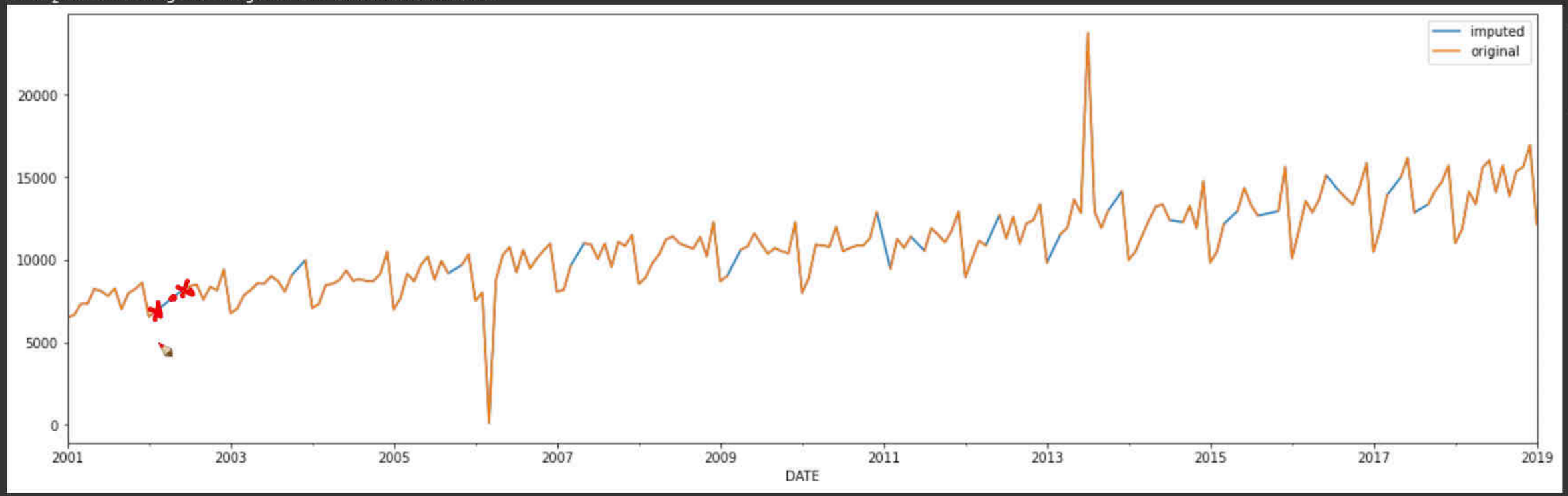
```
<matplotlib.legend.Legend at 0x7ff0b0c49750>
```



Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: An In | Time-Series.ipynb - Colab | Stationary process - Wiki | pandas.DataFrame.rolling | Partial autocorrelation fun | Exponential smoothing - W | Prophet | Forecasting at s | Forecasting at scale [Peer
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=KYHp72VMljcw
+ Code + Text
RAM
Disk

[13]
```
# linear interpolation
mobile_sales.Sales.interpolate(method='linear').plot(label='imputed')
mobile_sales.Sales.plot(label='original')
plt.legend()
```

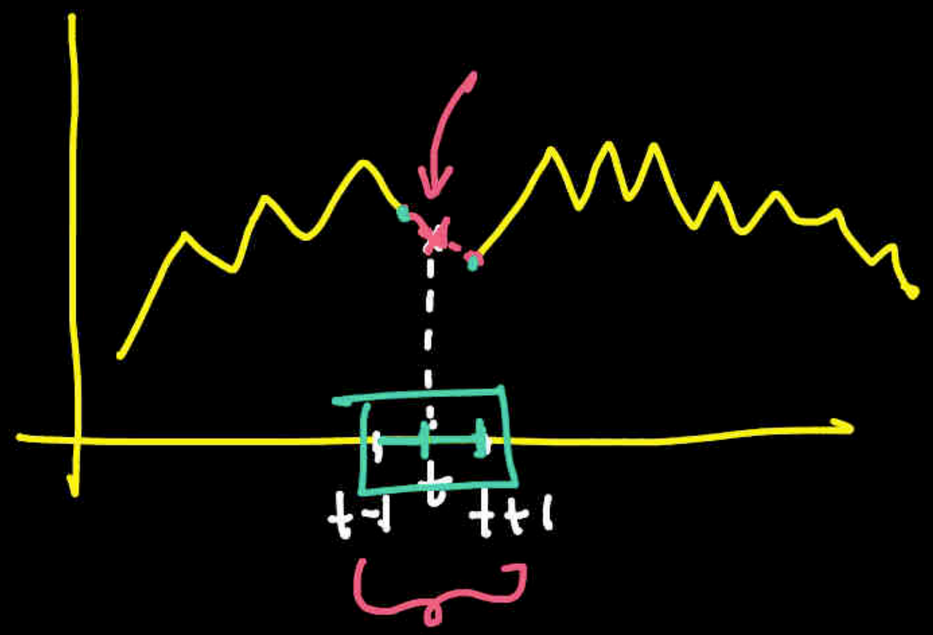
```
<matplotlib.legend.Legend at 0x7ff0b07116d0>
```



[15]
```
mobile_sales.Sales = mobile_sales.Sales.interpolate(method='linear')
mobile_sales.Sales.plot()
```

13 / 13

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:28



Linear-interpolati

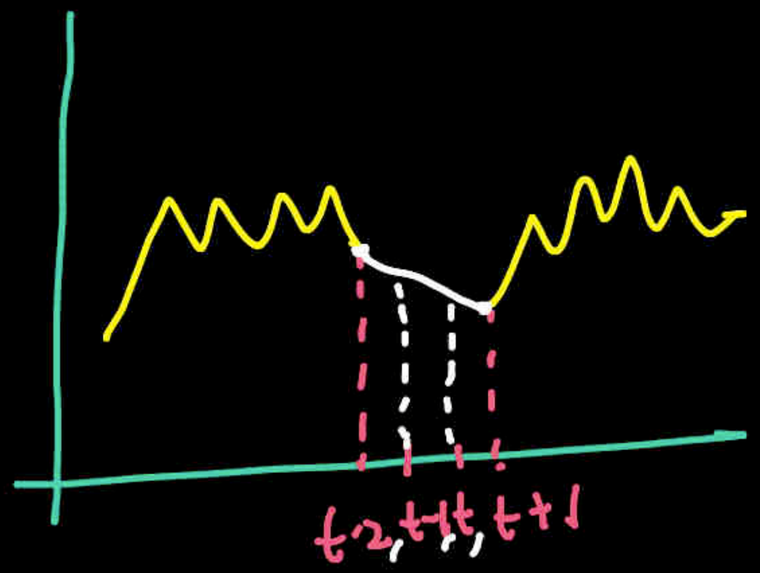
↓

CMA with m=1

centered moving average

14 / 15

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:29



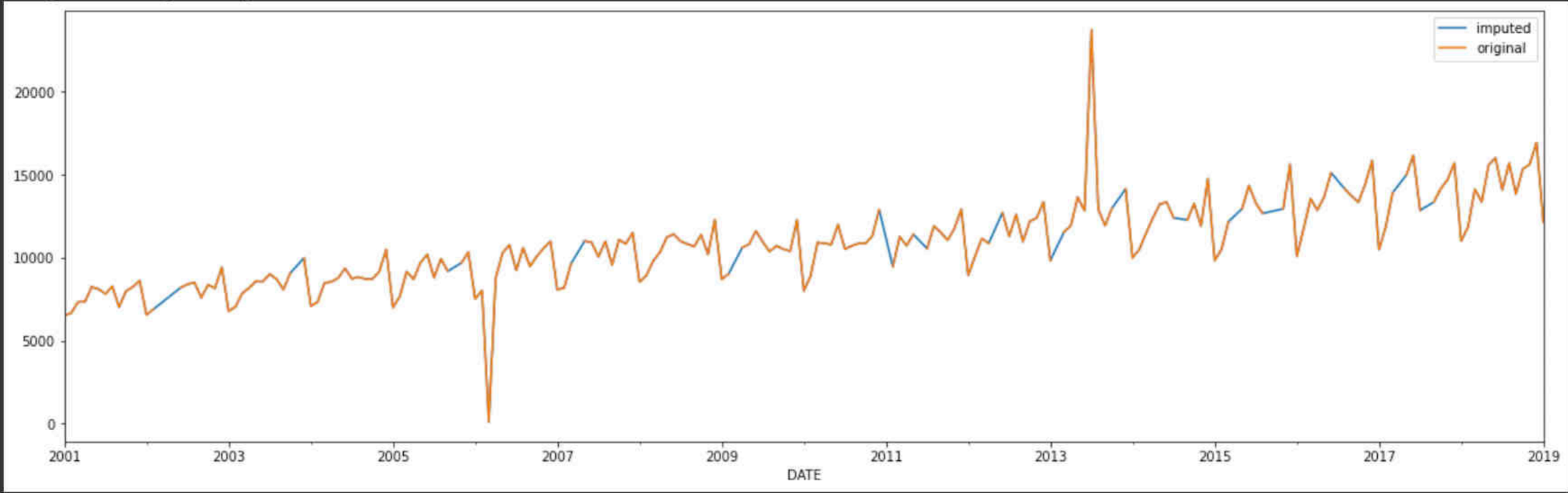
Linear - interpolation

15 / 15

```
# linear interpolation
mobile_sales.Sales.interpolate(method='linear').plot(label='imputed')
mobile_sales.Sales.plot(label='original')
plt.legend()
```

CMA · m=1

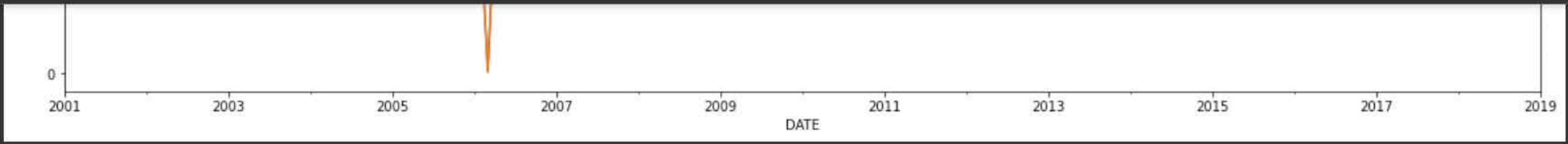
```
<matplotlib.legend.Legend at 0x7ff0b07116d0>
```




```
[15] mobile_sales.Sales = mobile_sales.Sales.interpolate(method='linear')
     mobile_sales.Sales.plot()
```

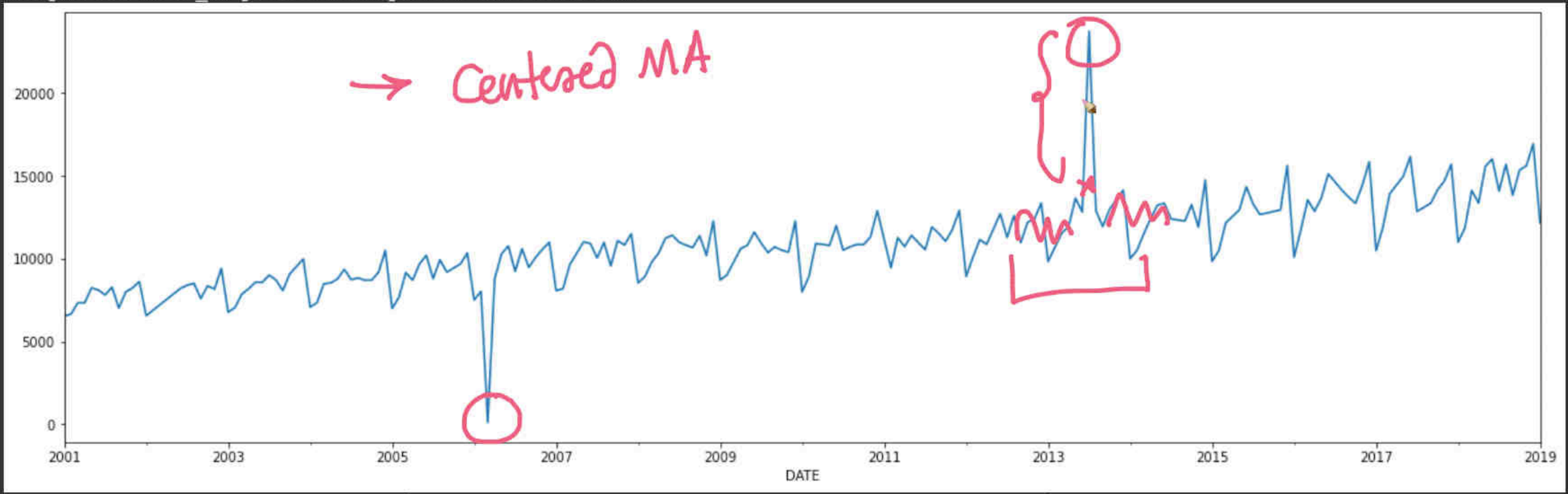
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelatio | Exponential smoothin | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=lrPzyAWQJCqm
+ Code + Text
RAM
Disk

[13]



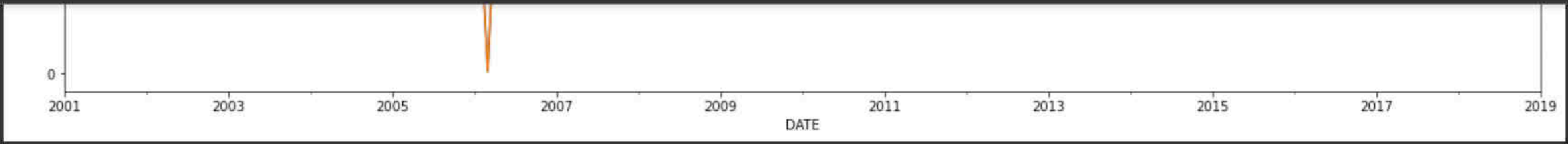

```
mobile_sales.Sales = mobile_sales.Sales.interpolate(method='linear')
mobile_sales.Sales.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0aedcd4d0>
```



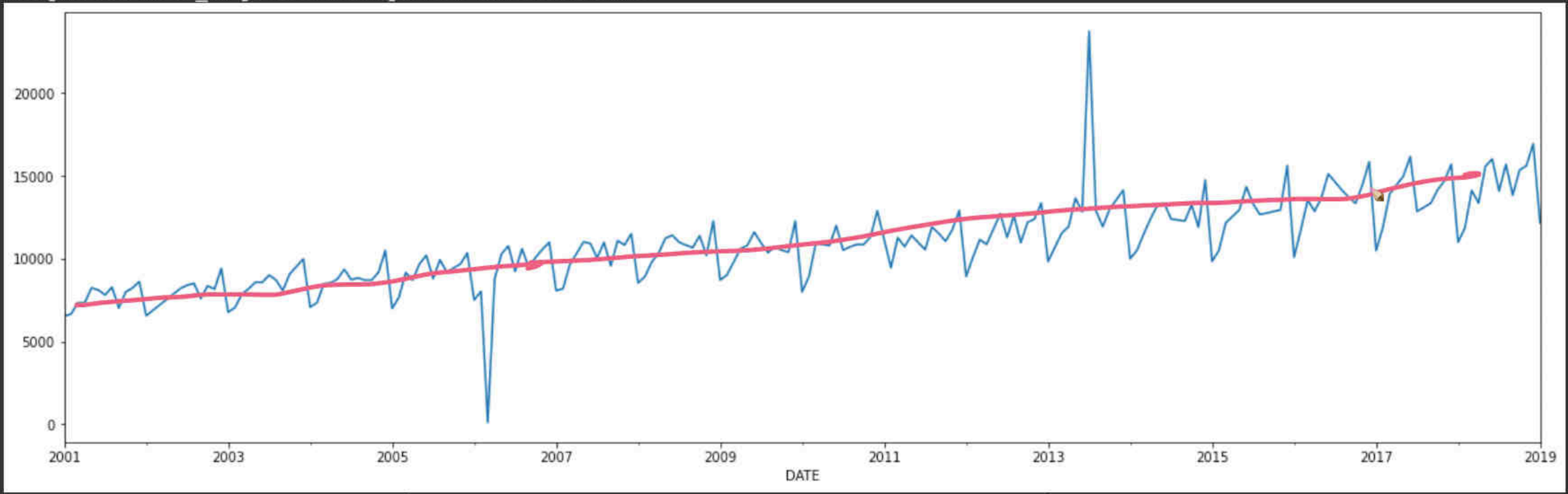
17 / 17

[13]



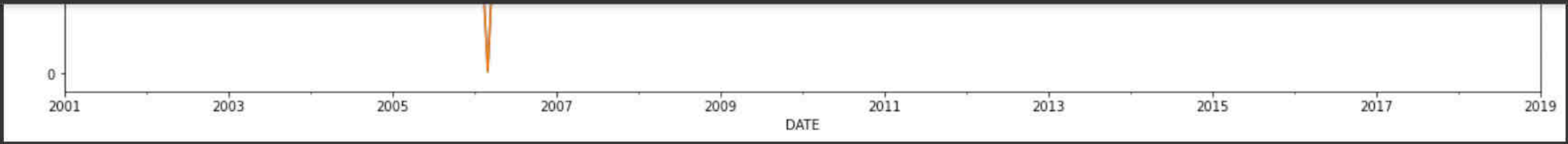

```
mobile_sales.Sales = mobile_sales.Sales.interpolate(method='linear')
mobile_sales.Sales.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0aedcd4d0>
```



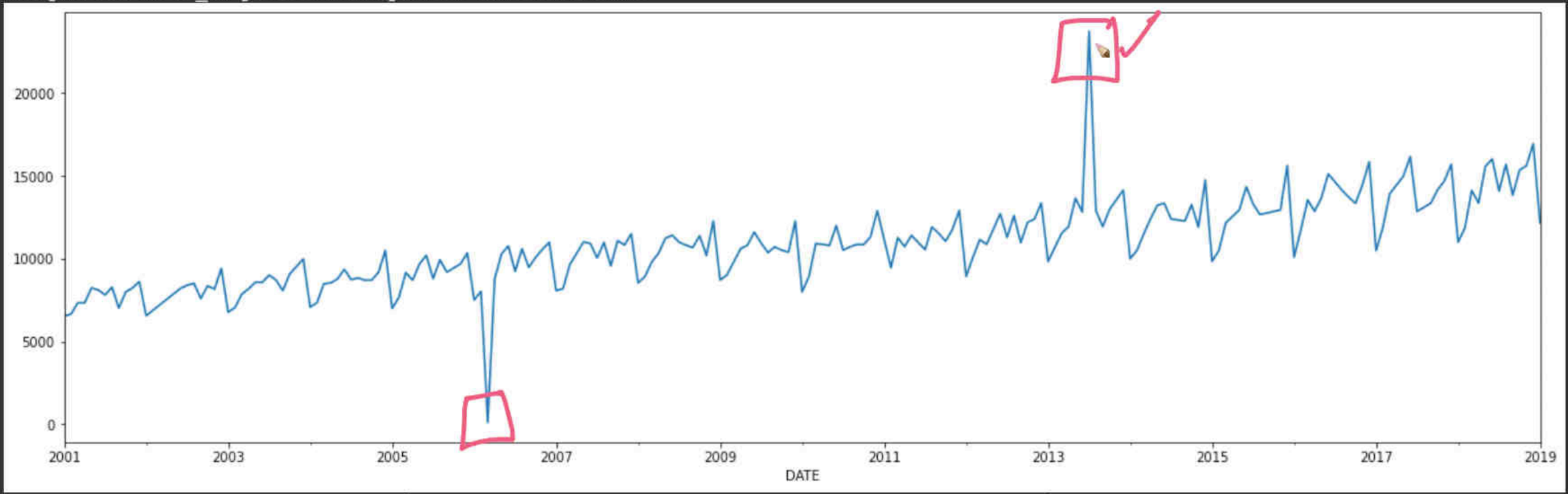
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=lrPzyAWQJCqm
+ Code + Text
RAM
Disk

[13]




```
mobile_sales.Sales = mobile_sales.Sales.interpolate(method='linear')
mobile_sales.Sales.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0aedcd4d0>
```



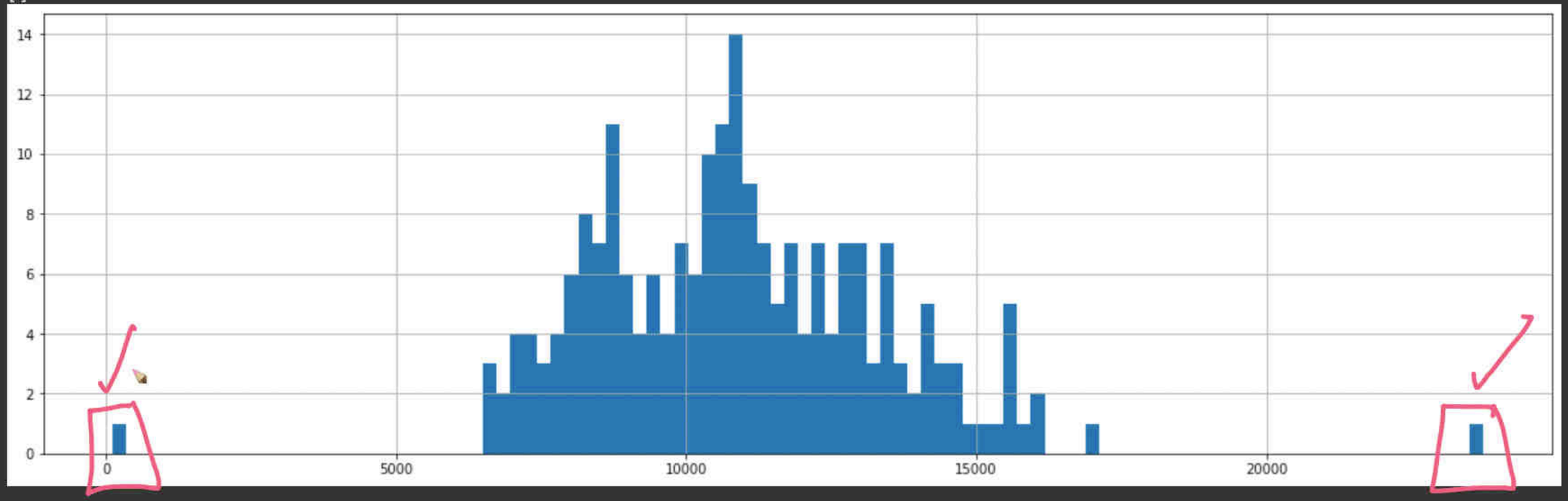
19 / 19

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelatio | Exponential smoothin | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=lrPzyAWQJCqm
+ Code + Text
RAM
Disk

DATE

```
[16] # Remove anamolies using percentiles
     mobile_sales.Sales.hist(bins=100).plot()
```

[]




```
[17] mobile_sales.Sales.describe()
```

```
count      217.000000
mean     10873.046083
std       2571.620954
```

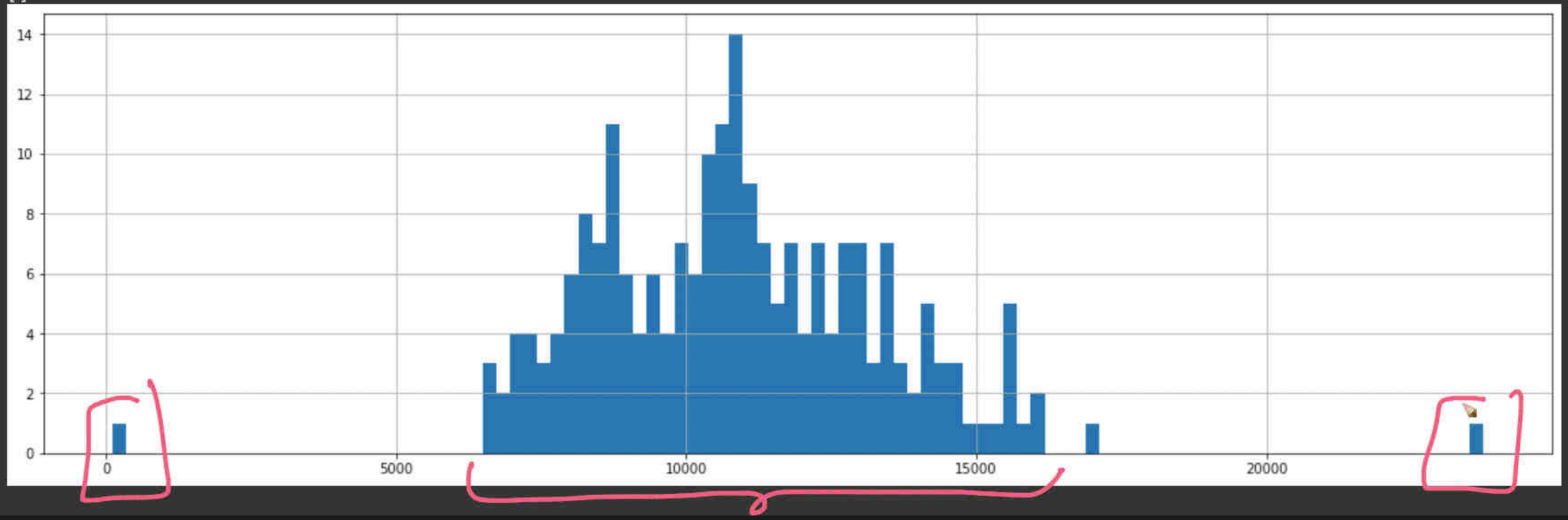
20 / 20

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:36
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=lrPzyAWQJCqm
+ Code + Text
RAM
Disk

DATE

```
[16] # Remove anamolies using percentiles
     mobile_sales.Sales.hist(bins=100).plot()
```

[]




```
[17] mobile_sales.Sales.describe()
```

```
count      217.000000
mean     10873.046083
std       2571.620954
```

21 / 21

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:41

Simple MA:

$$\hat{x}_t = \frac{x_{t-1} + x_{t-2} + x_{t-3} + \cdots + x_{t-m}}{m}$$

Weighted MA:

$$\hat{x}_t = \frac{\sum_{i=1}^{m} \alpha_i \, x_{t-i}}{\sum_{i=1}^{m} \alpha_i}$$

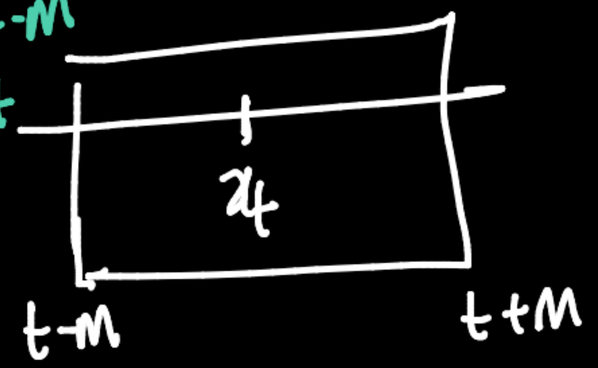
remove outliers ← Centered MA

$$\hat{x}_t = \frac{1}{2m+1} \sum_{i=t-m}^{t+m} x_i$$

$$\underbrace{x_{t-m} \cdots}_{m} \; \underset{1}{\boxed{x_t}} \; \underbrace{\cdots x_{t+m}}_{m}$$

Alt:

$$\hat{x}_t = \frac{1}{2m} \sum_{\substack{i=t-m \\ i \neq t}}^{t+m} x_i$$



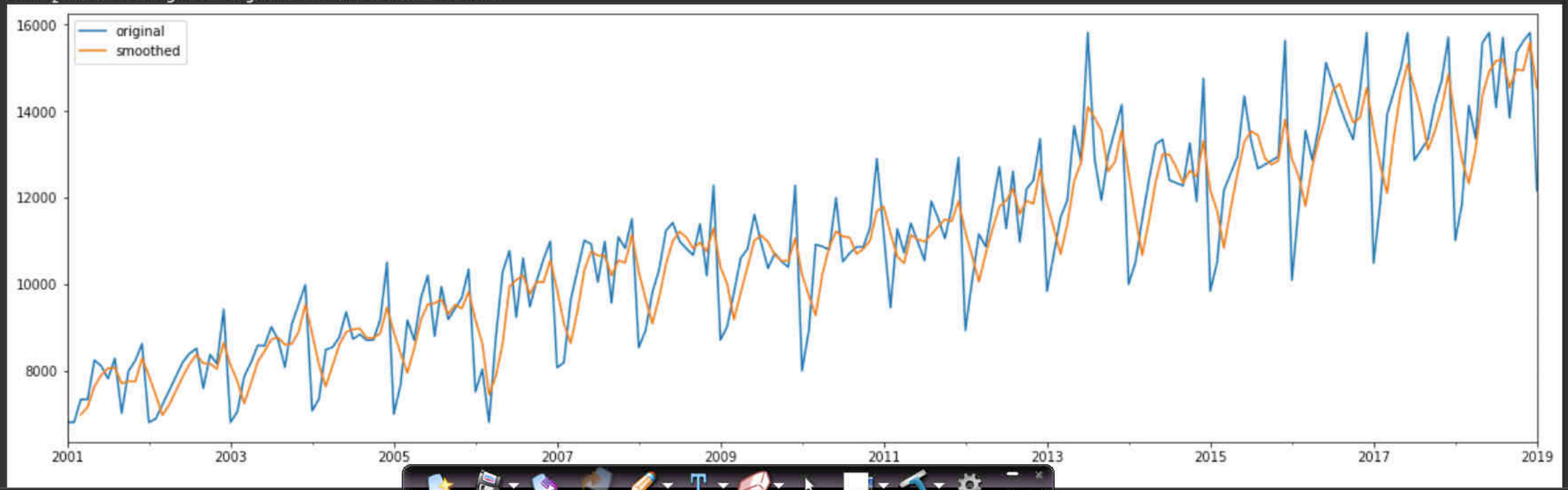
Moving Averages

```python
# Plot original data
mobile_sales.Sales.plot(label='original')

# Plot data after implementing Moving Averages
mobile_sales.Sales.rolling(3, center=False).mean().plot(label='smoothed')

plt.legend()
```

NOT Centered MA

$$\hat{x}_t = \frac{x_{t-1} + x_{t-2} + x_{t-3}}{3}$$

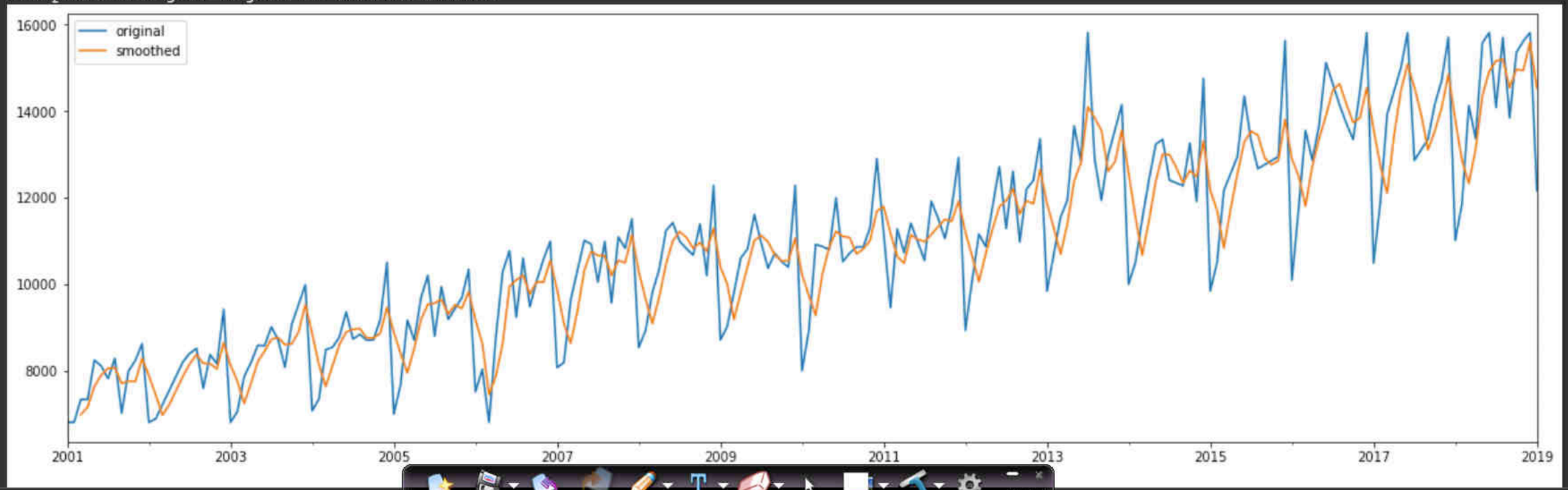
```
<matplotlib.legend.Legend at 0x7ff0aea161d0>
```



Moving Averages

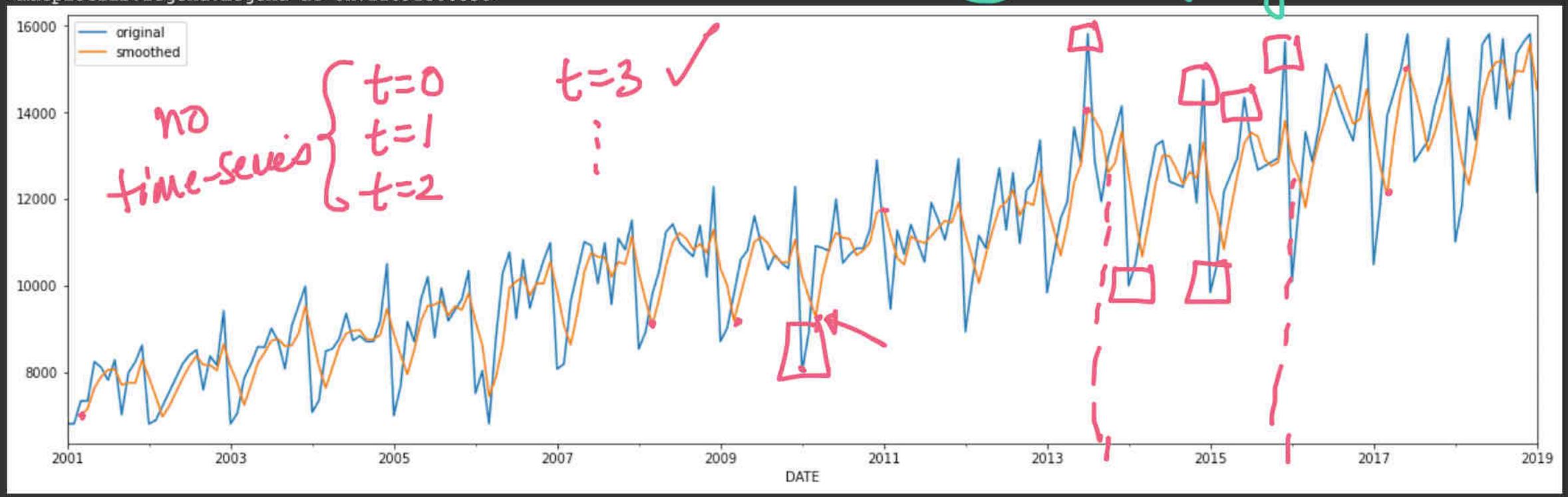
```
# Plot original data
mobile_sales.Sales.plot(label='original')

# Plot data after implementing Moving Averages
mobile_sales.Sales.rolling(3, center=False).mean().plot(label='smoothed')

plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ff0aea161d0>
```



```
# Plot original data
mobile_sales.Sales.plot(label='original')

# Plot data after implementing Moving Averages
mobile_sales.Sales.rolling(3, center=False).mean().plot(label='smoothed')

plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ff092806850>
```




```
[24] # Plot original data
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in

colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=y1mGQ_SsMgwt

+ Code + Text

RAM Disk

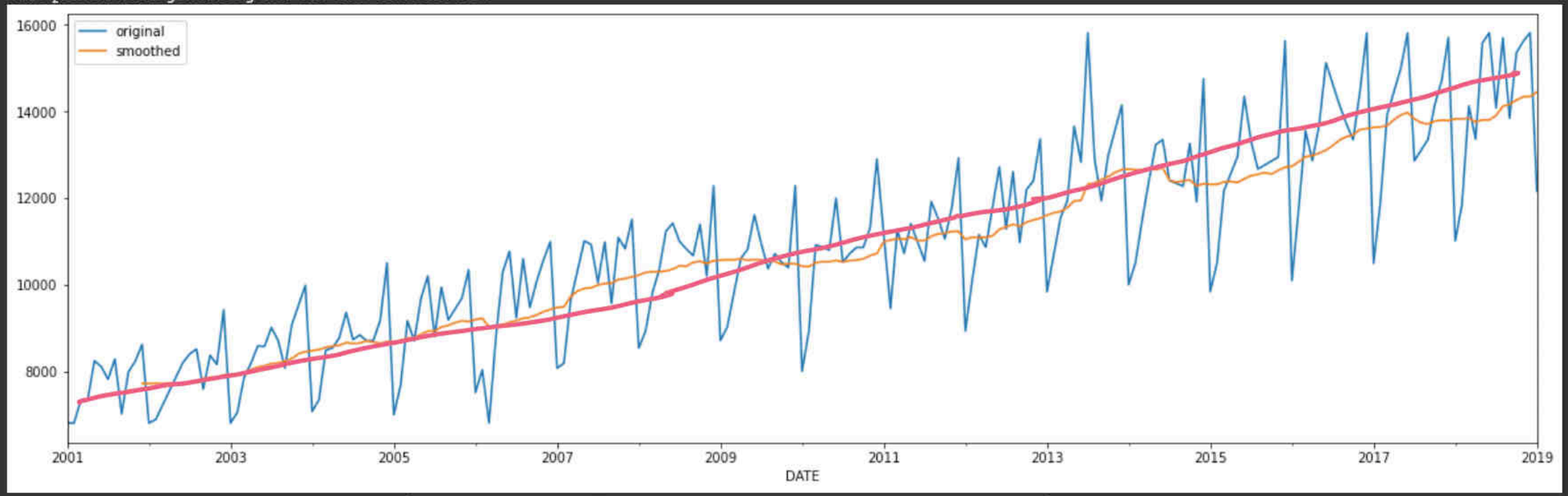
Decompose a time-series

Linear regr

x_i : month-num
y_i : sales

```
#Trend line: [Which ML technique can we use?]
mobile_sales.Sales.plot(label='original')
mobile_sales.Sales.rolling(12, center=False).mean().plot(label='smoothed')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ff0ae821950>
```



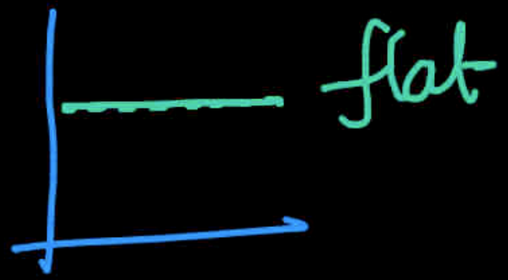
27 / 29

Q

{ Moving average → sub-optimal solution

Trend:

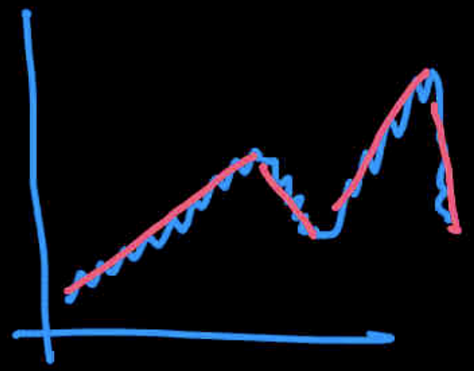
(INTUITION)



flat

incr

decr



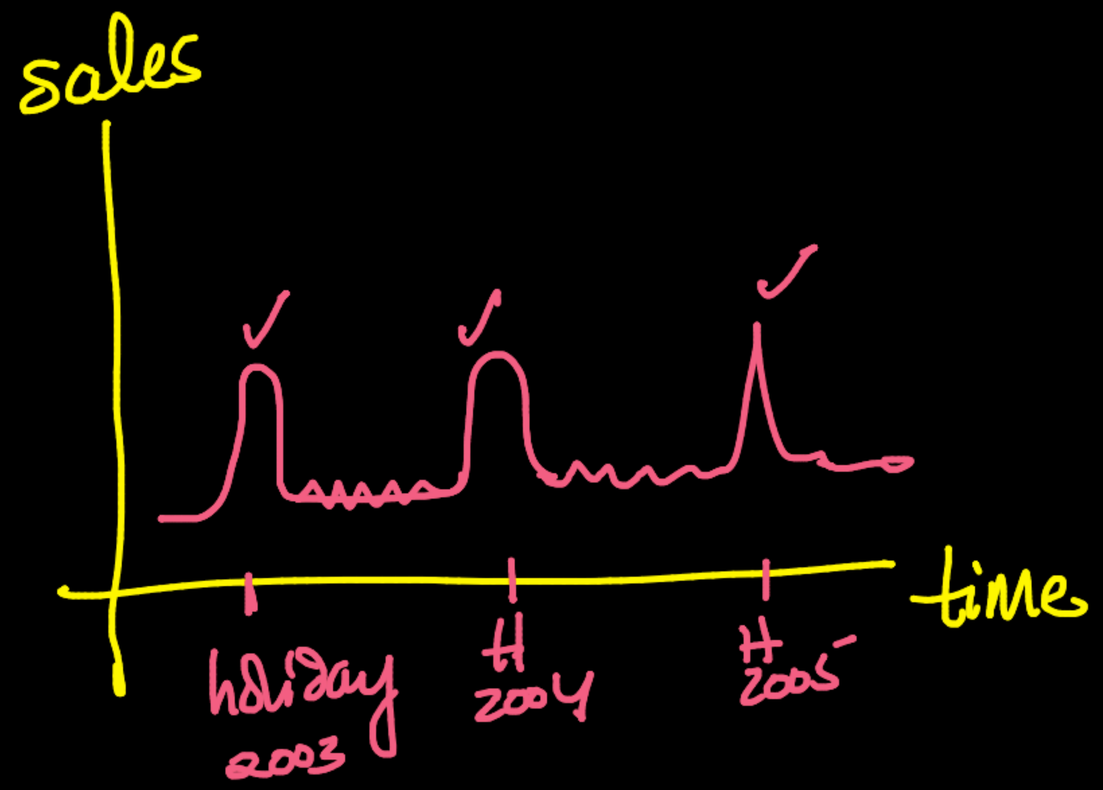
changing

Chrome File Edit View History Bookmarks Profiles Tab Window Help
29 / 31

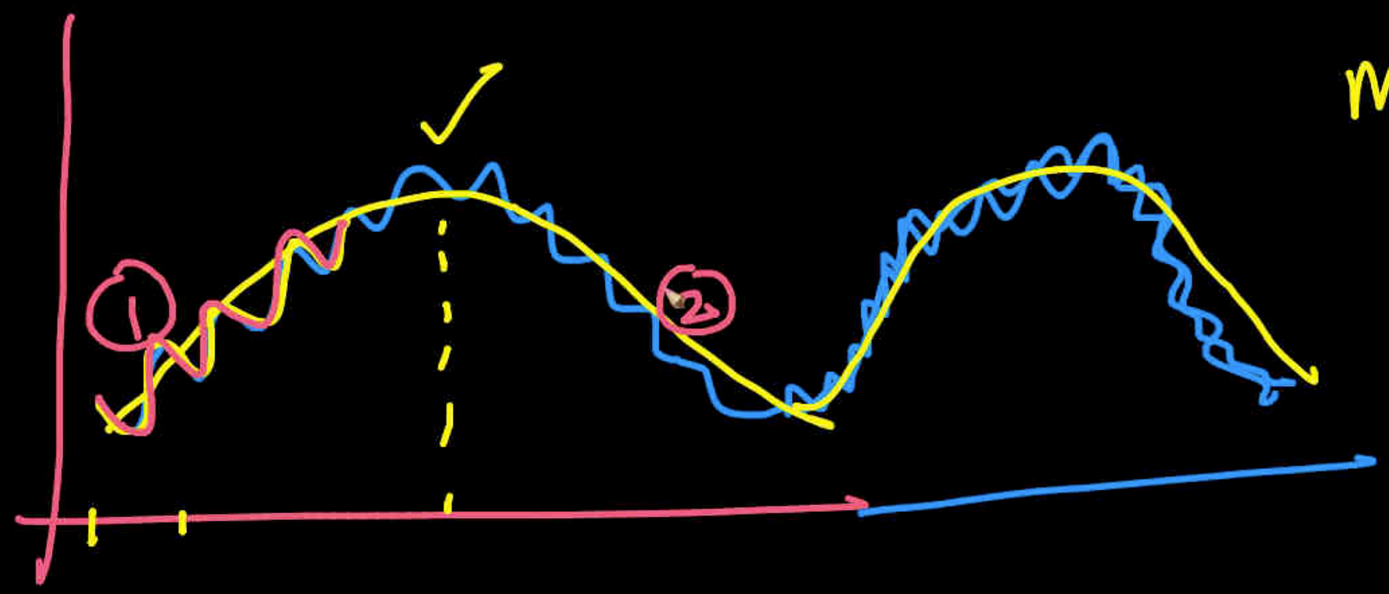
Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:59

Seasonality

(INTUITION)



30 / 31



more than one seasonality

Time-series

Sales

$$x(t) = b(t) + s(t) + e(t)$$

trend

Seasonality

error/noise

Additive TS-decomposition

Regression: $y_i = w_0 + w_1 x_{i1} + \cdots + w_d x_{id} + \epsilon_i$

irreducible error

$$e(t) = x(t) - [b(t) + s(t)]$$

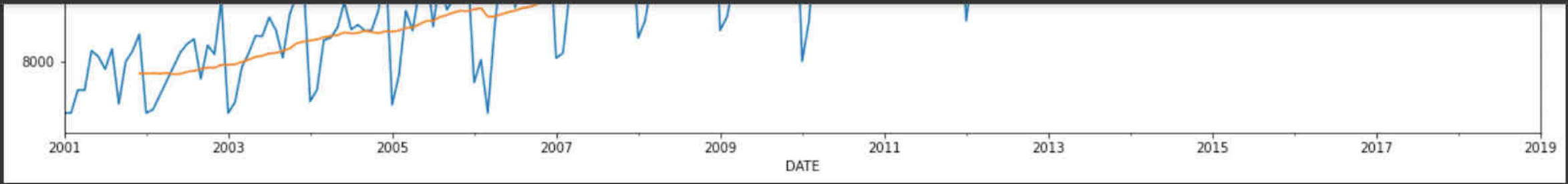
$e_0, e_1, e_2 \ldots\ldots e_t \rightarrow$ dist of errors

ideally $\sim N(0, \sigma^2)$

small

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:07
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=7PnD1qm7Mwnz
+ Code + Text
RAM
Disk

[26]



issues versions

```
#additive decomposition
import statsmodels.api as sm

model = sm.tsa.seasonal_decompose(mobile_sales.Sales, model='additive')
```

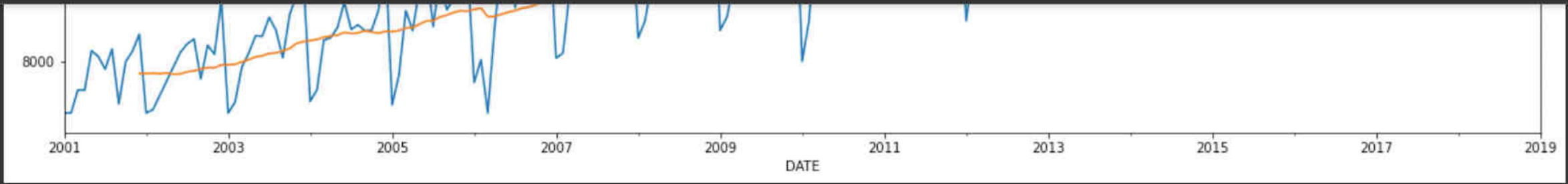
```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functio
  import pandas.util.testing as tm
```

```
# By default, we get the plot twice with this functionality
# We add ; to avoid seeing the plot

model.plot();
```

34 / 34

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:08
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=7PnD1qm7Mwnz
+ Code + Text
RAM
Disk

[26]




```
#additive decomposition
import statsmodels.api as sm

model = sm.tsa.seasonal_decompose(mobile_sales.Sales, model='additive')
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functio
  import pandas.util.testing as tm
```

```
# By default, we get the plot twice with this functionality
# We add ; to avoid seeing the plot

model.plot();
```

35 / 35

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in

colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=BXcxfVCvM7cA

+ Code + Text

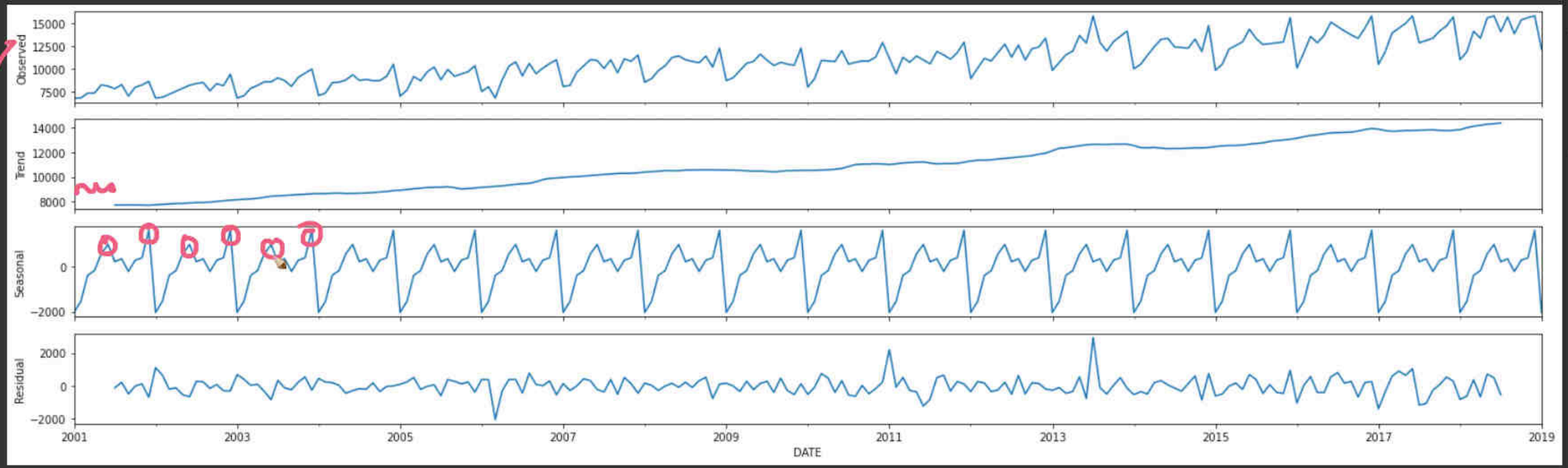
RAM Disk

```
import pandas.util.testing as tm
```

MA เข้าไป
ใช้ moving average

```
# By default, we get the plot twice with this functionality
# We add ; to avoid seeing the plot

model.plot();
```

$$x(t) = b(t) + s(t) + e(t)$$




```
model.resid.plot()
```

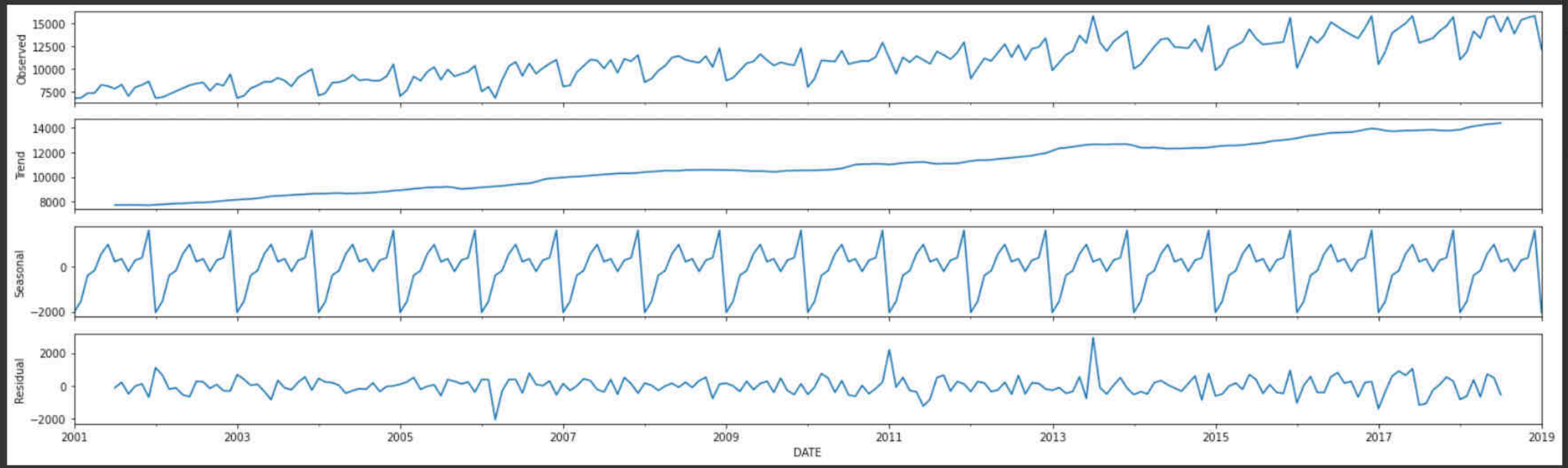
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0965624d0>
```

36 / 36

```
import pandas.util.testing as tm
```

```
# By default, we get the plot twice with this functionality
# We add ; to avoid seeing the plot

model.plot();
```

Fourier-decomp

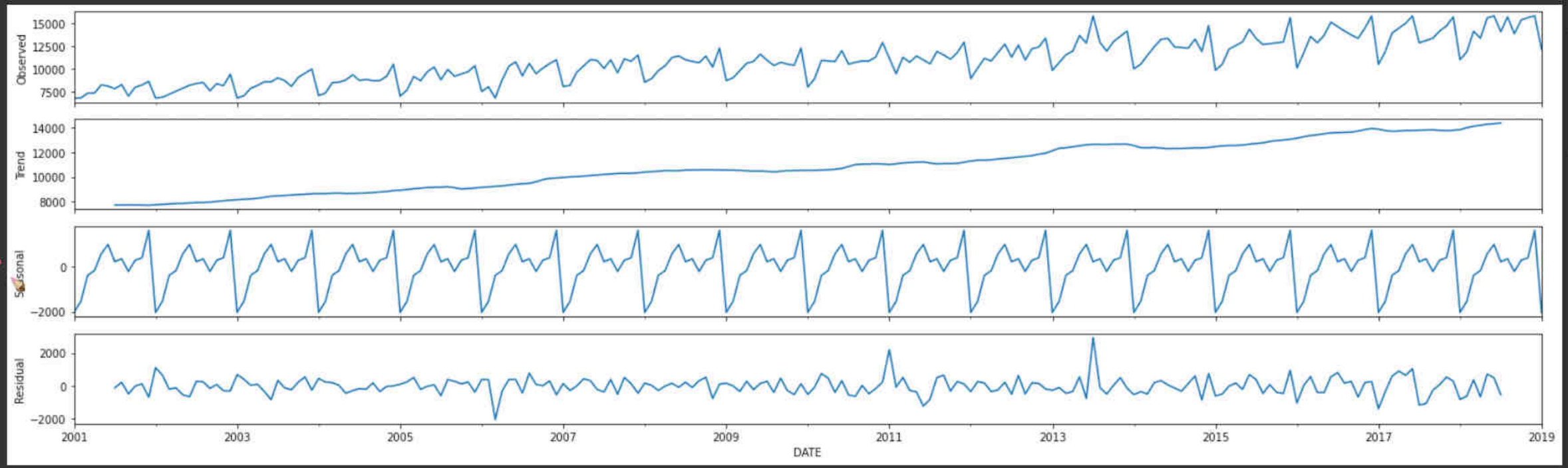



```
model.resid.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0965624d0>
```

```
import pandas.util.testing as tm
```

```
# By default, we get the plot twice with this functionality
# We add ; to avoid seeing the plot

model.plot();
```




```
model.resid.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0965624d0>
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Forecasting at Uber: | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelatio | Exponential smoothin | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in

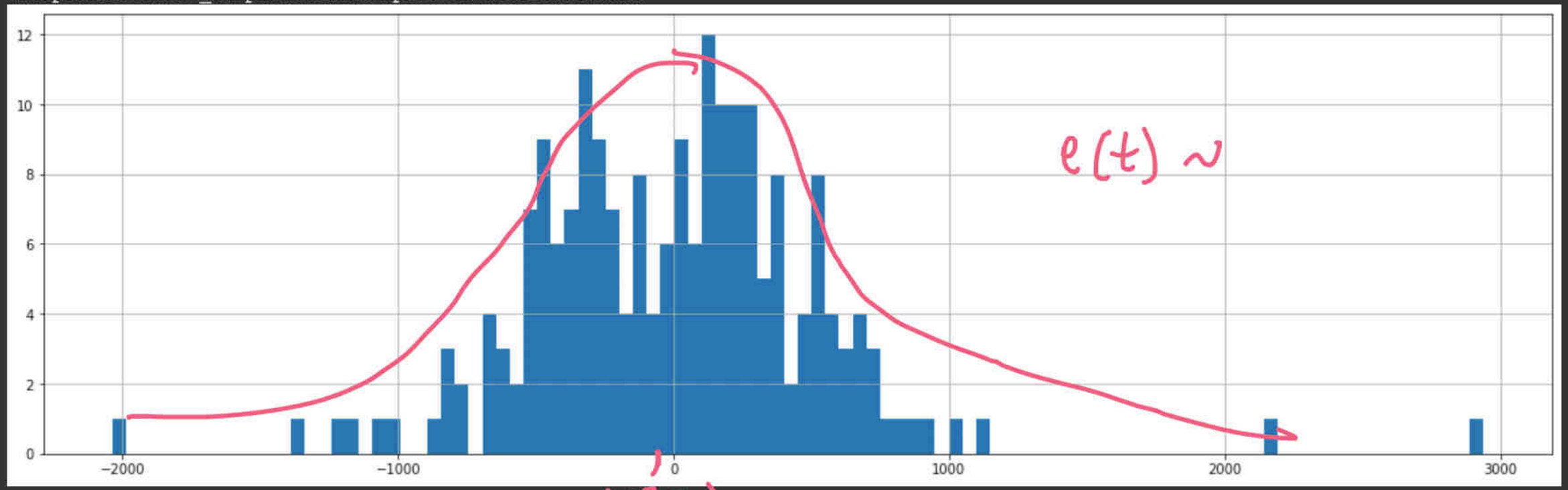
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=BXcxfVCvM7cA

+ Code + Text

RAM Disk

[30]
```
model.resid.hist(bins=100)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff096477050>
```



(-3.2)

[31]
```
model.resid.mean()
```

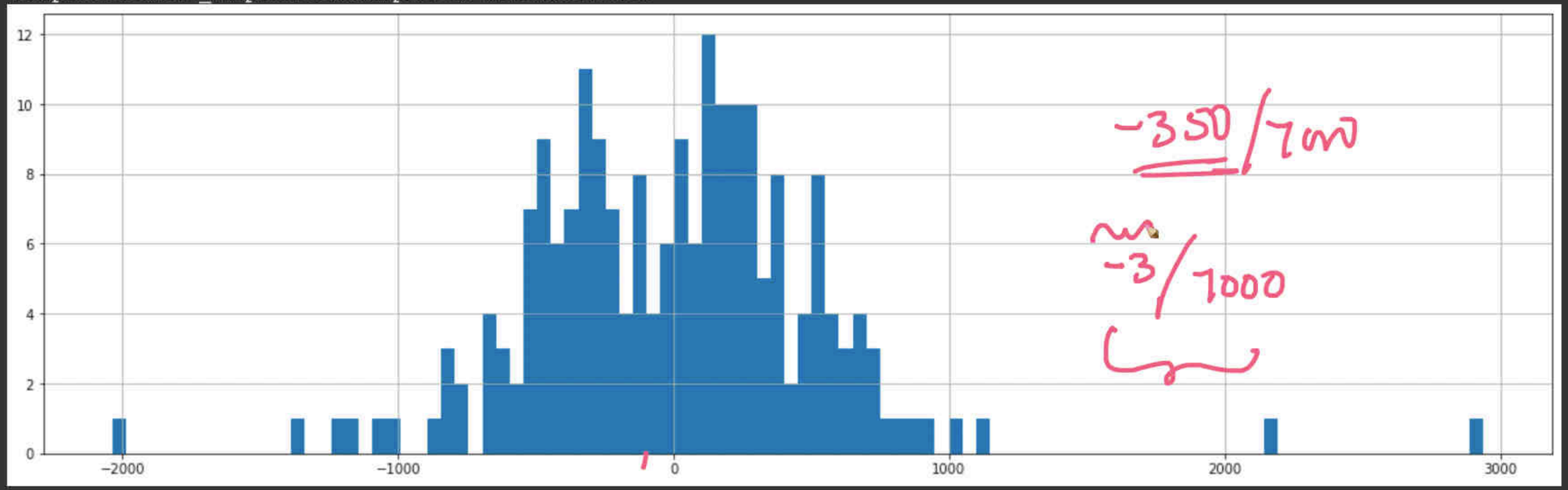
```
-3.2316417029771514
```

Stationarity

39 / 39

[30]

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff096477050>
```



[31]

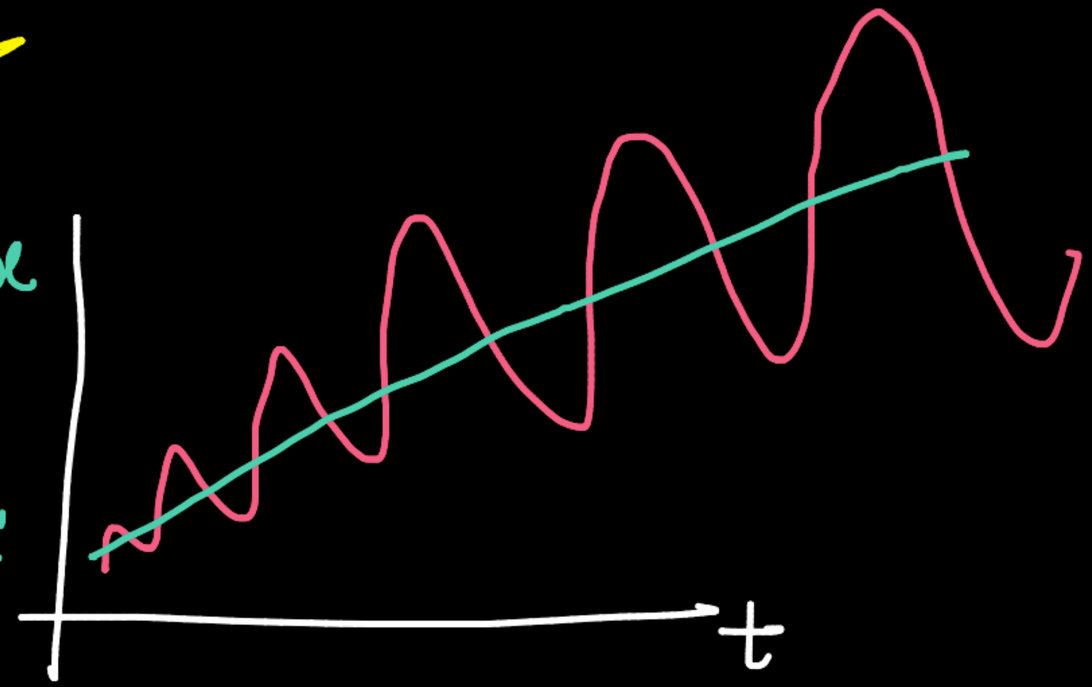
```
model.resid.mean()
```

```
-3.2316417029771514
```

Stationarity

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:18

Multiplicative Decomp



example:

$$\hat{x}(t) = b(t)\, s(t)$$

$$\{\; x(t) = b(t) \cdot s(t) \cdot e(t)$$

log-transforms

Additive

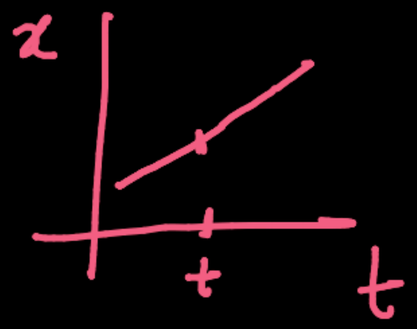
$$\log(x(t)) = \log(b(t)) + \log(s(t)) + \log(e(t))$$

41 / 42

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:23

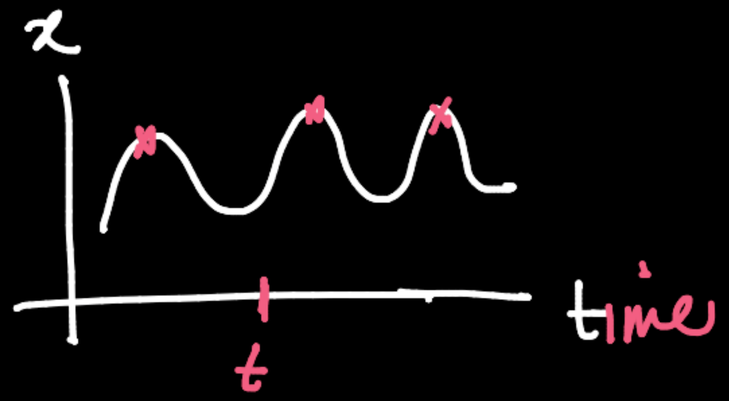
Stationary TS

params (mean, var, std-dev, amplitude, freq) of a time-series are not dependent on time



Trend → non-stationary

Seasonality → " "

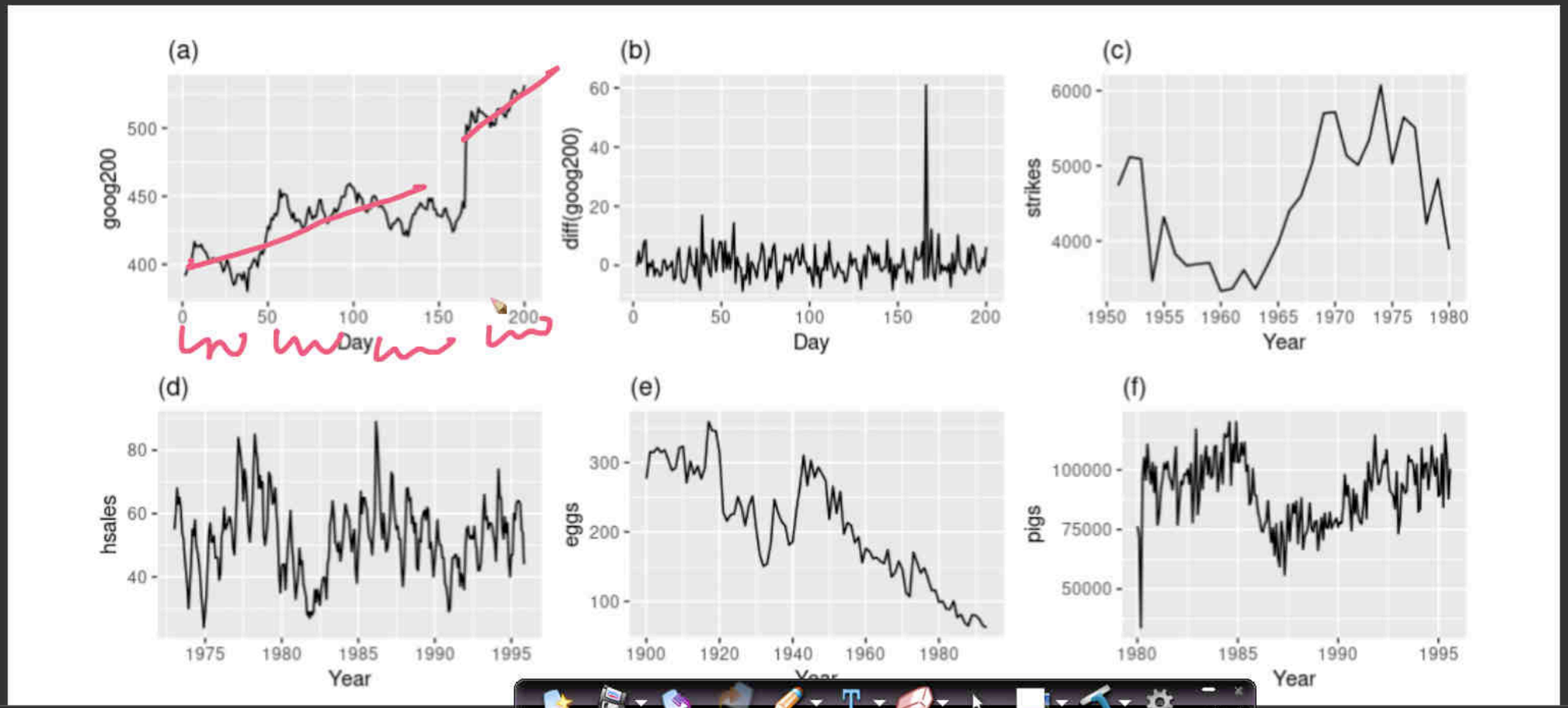


42 / 42

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=9yf9_JOINLRm
+ Code + Text
RAM
Disk

Stationarity

Q. Which of the following time series is stationary?



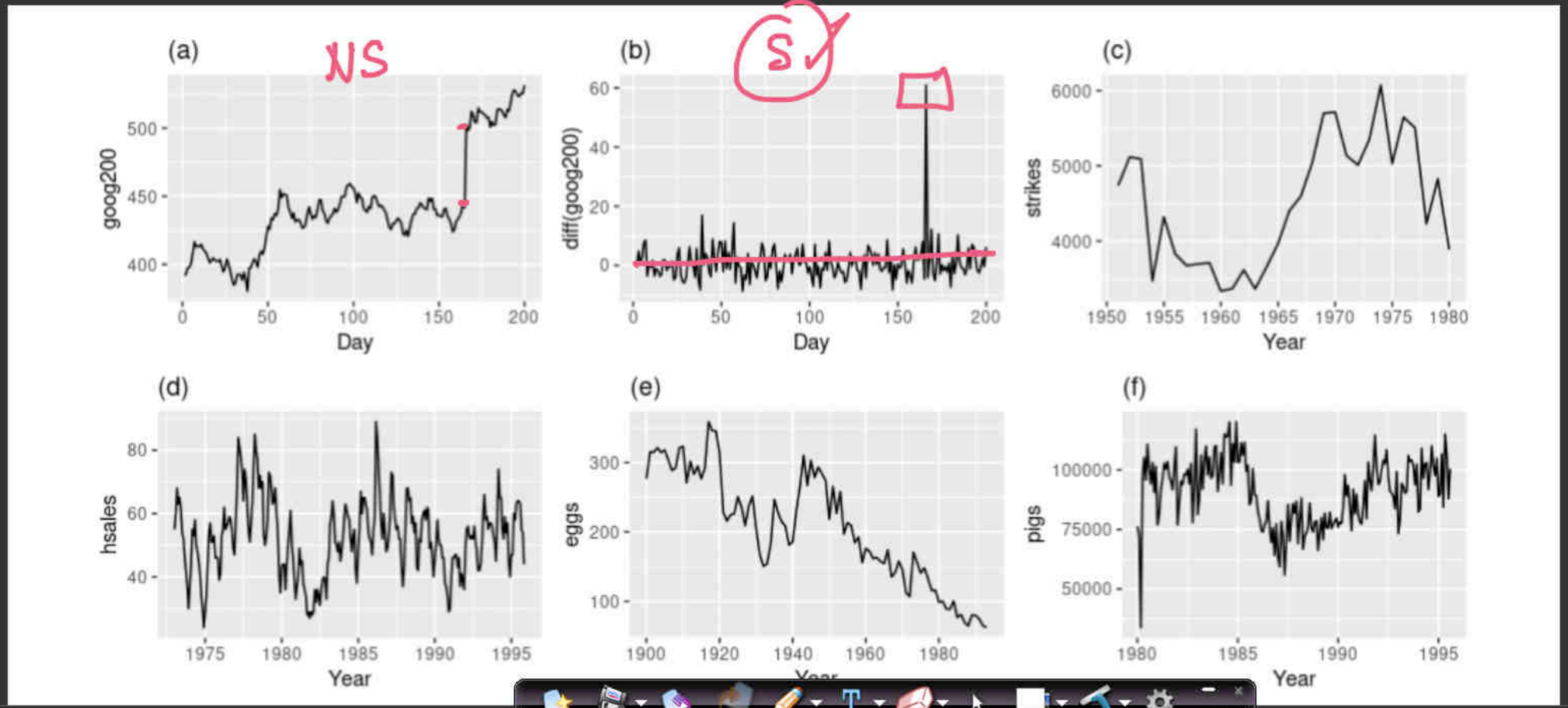
43 / 43

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=9yf9_JOINLRm
+ Code + Text
RAM
Disk

Stationarity

Noise → stationary

Q. Which of the following time series is stationary?

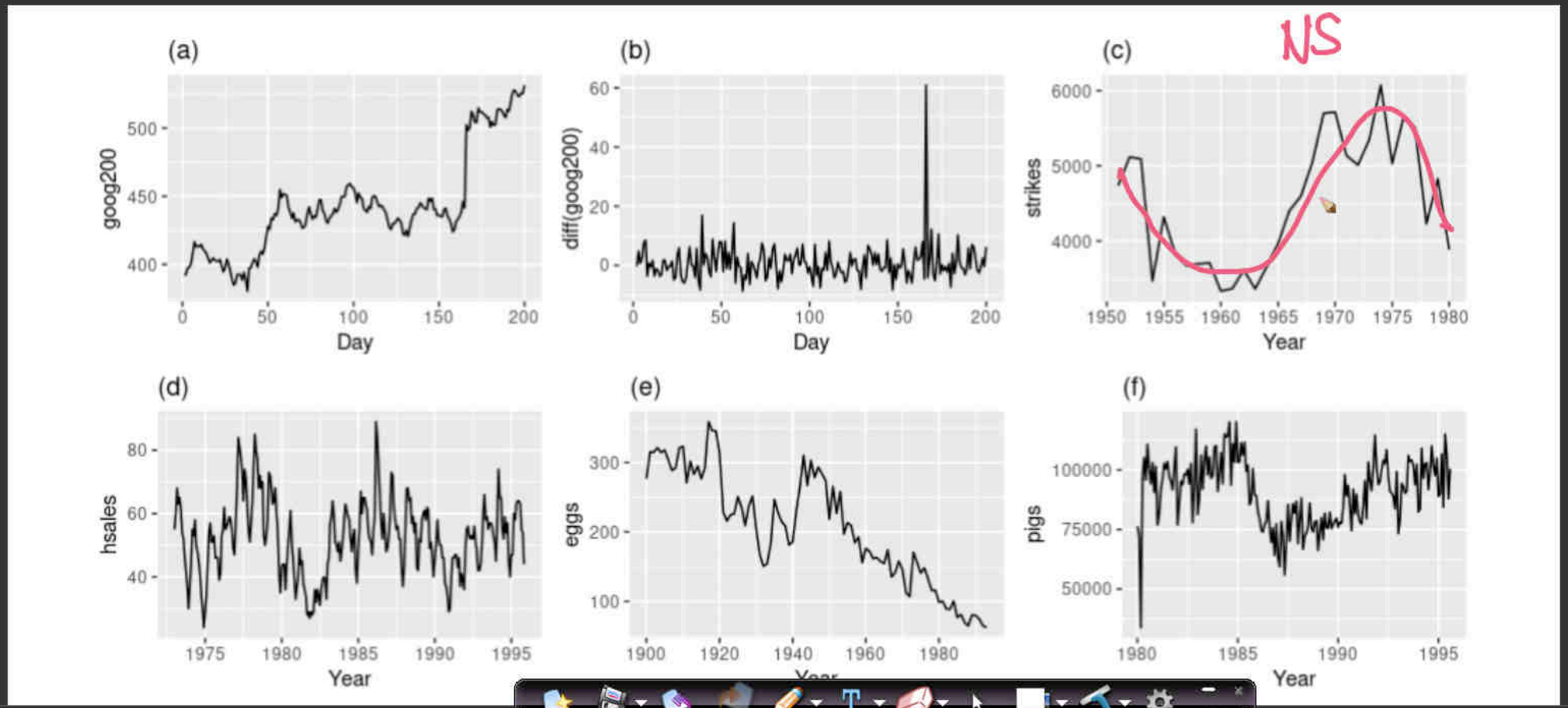


44 / 44

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=9yf9_JOINLRm
+ Code + Text
RAM
Disk

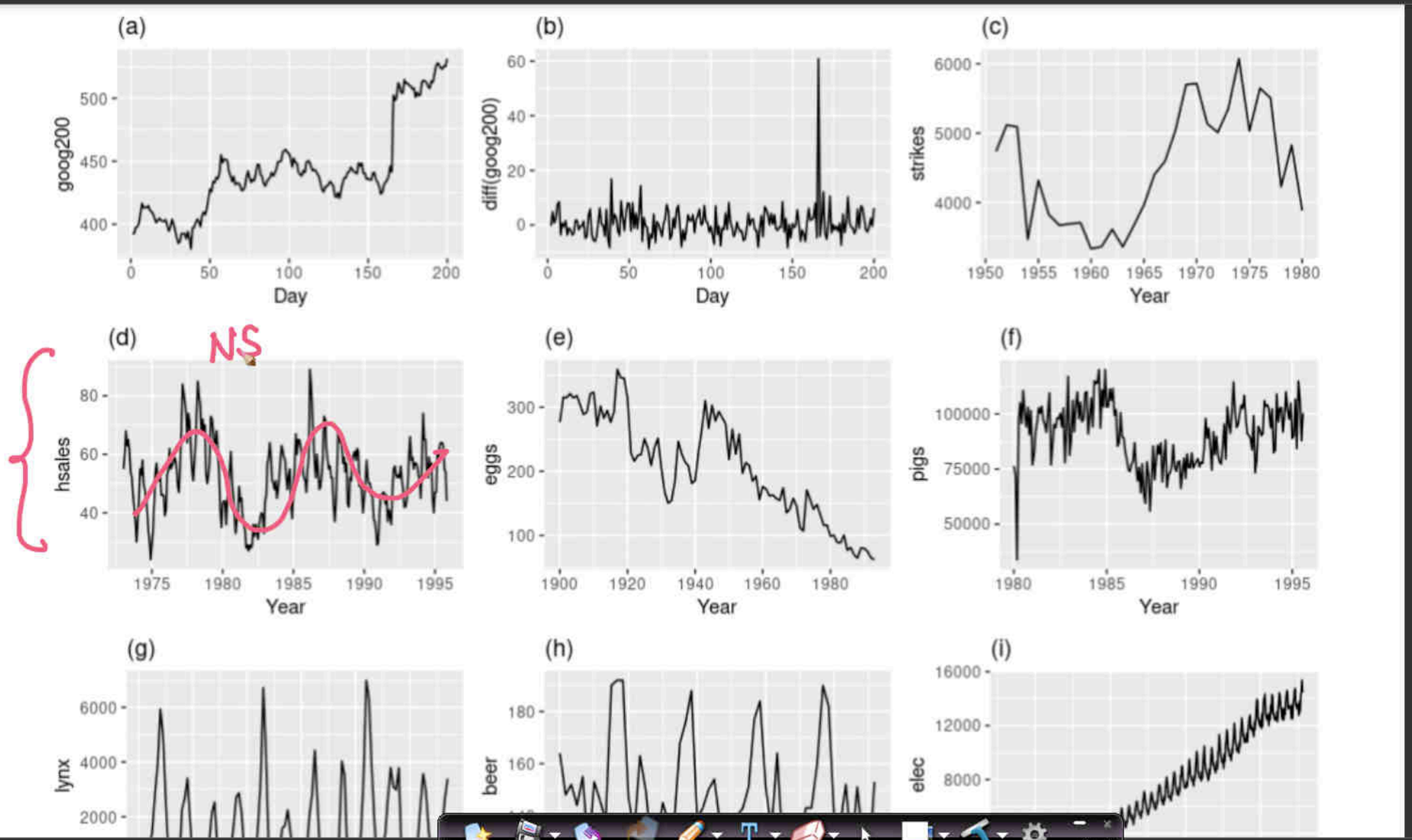
Stationarity

Q. Which of the following time series is stationary?

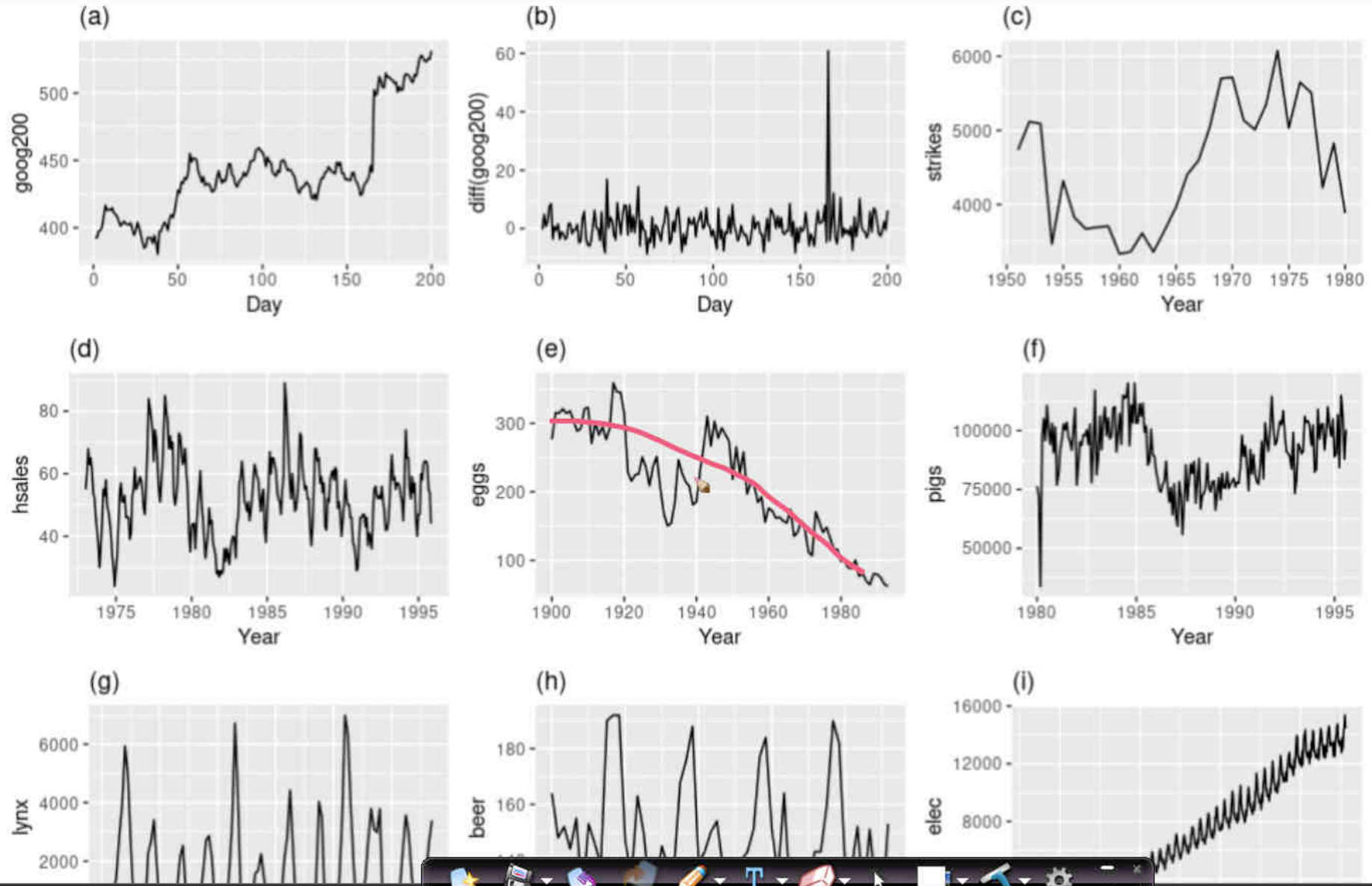


45 / 45

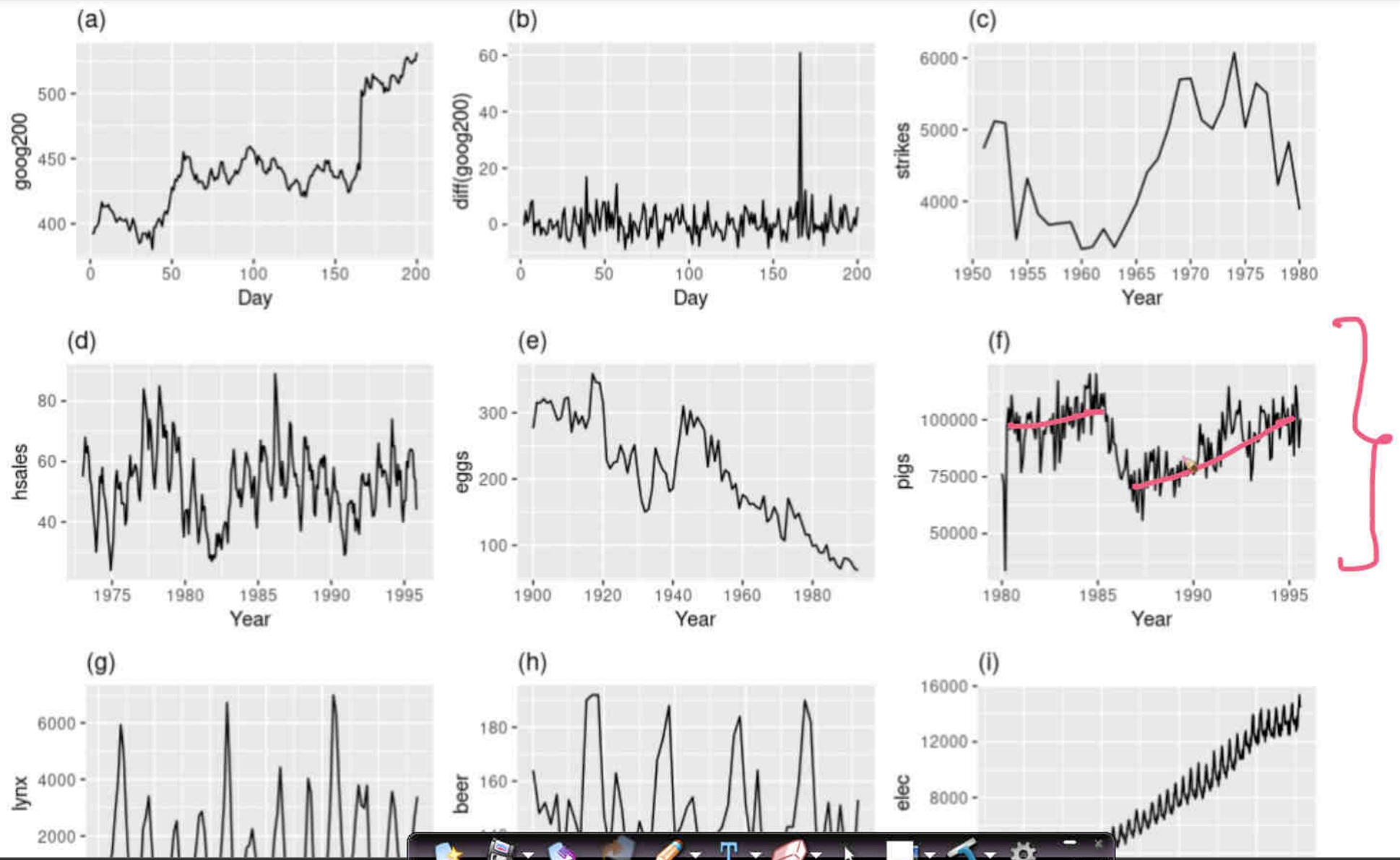
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=9yf9_JOINLRm
+ Code + Text
RAM
Disk



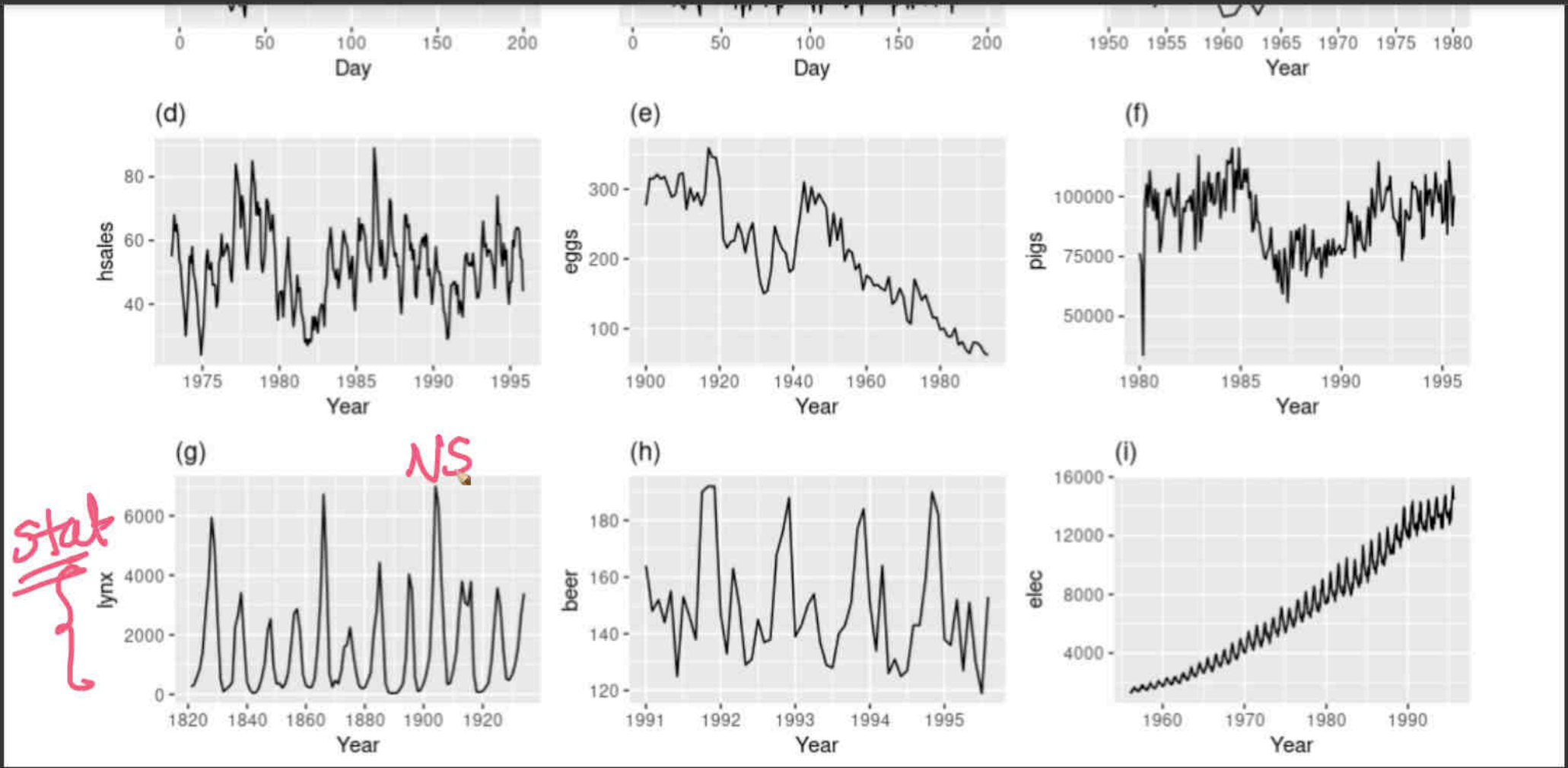
46 / 46


(a)
goog200
500
450
400
0
50
100
150
200
Day
(b)
diff(goog200)
60
40
20
0
0
50
100
150
200
Day
(c)
strikes
6000
5000
4000
1950
1955
1960
1965
1970
1975
1980
Year
(d)
hsales
80
60
40
1975
1980
1985
1990
1995
Year
(e)
eggs
300
200
100
1900
1920
1940
1960
1980
Year
(f)
pigs
100000
75000
50000
1980
1985
1990
1995
Year
(g)
lynx
6000
4000
2000
(h)
beer
180
160
(i)
elec
16000
12000
8000

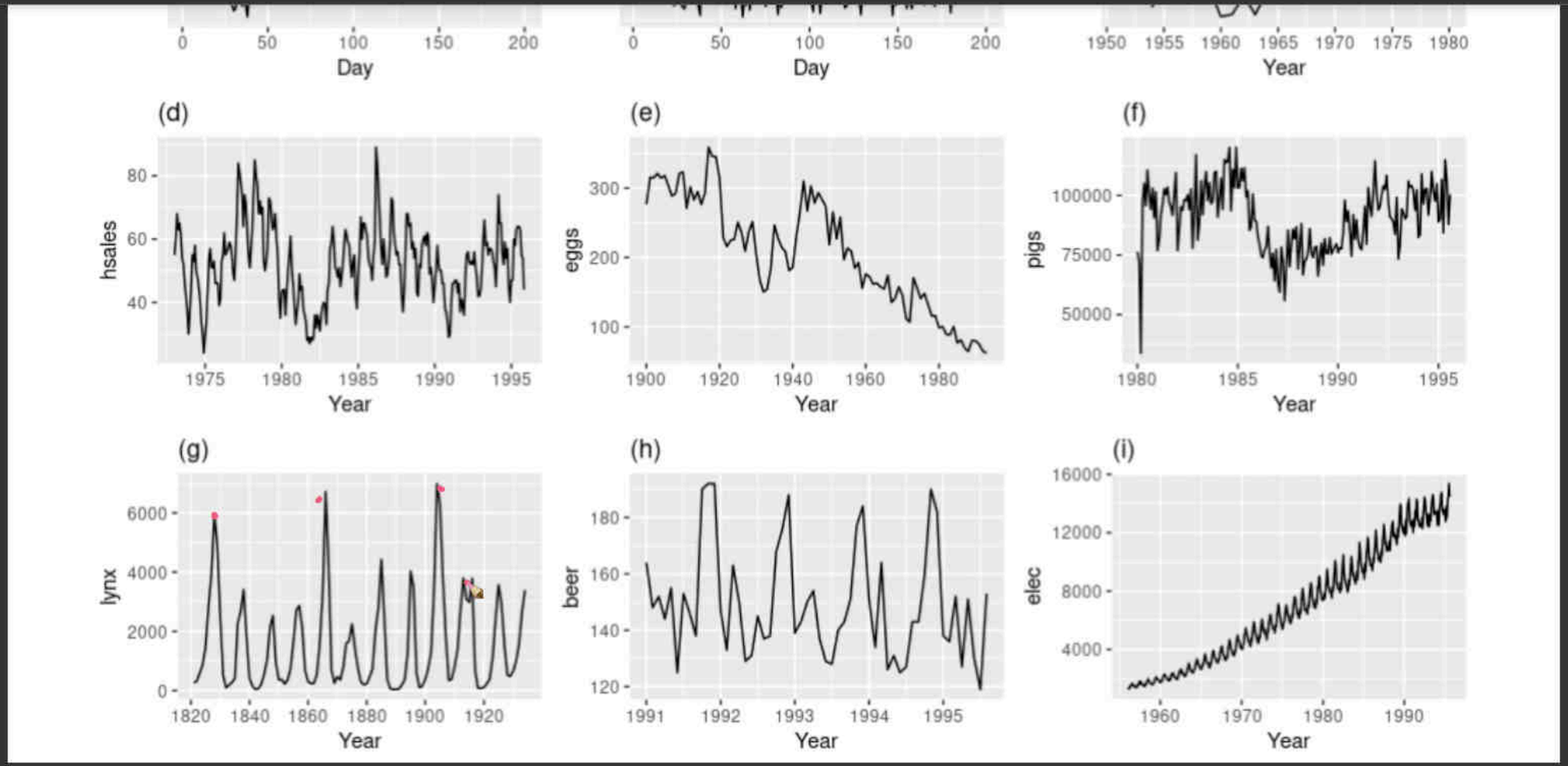
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting at Uber: A | Time-Series.ipynb - C | Stationary process - W | pandas.DataFrame.ro | Partial autocorrelation | Exponential smoothing | Prophet | Forecasting | Forecasting at scale | pandas.DataFrame.in
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=9yf9_JOINLRm
+ Code + Text
RAM
Disk



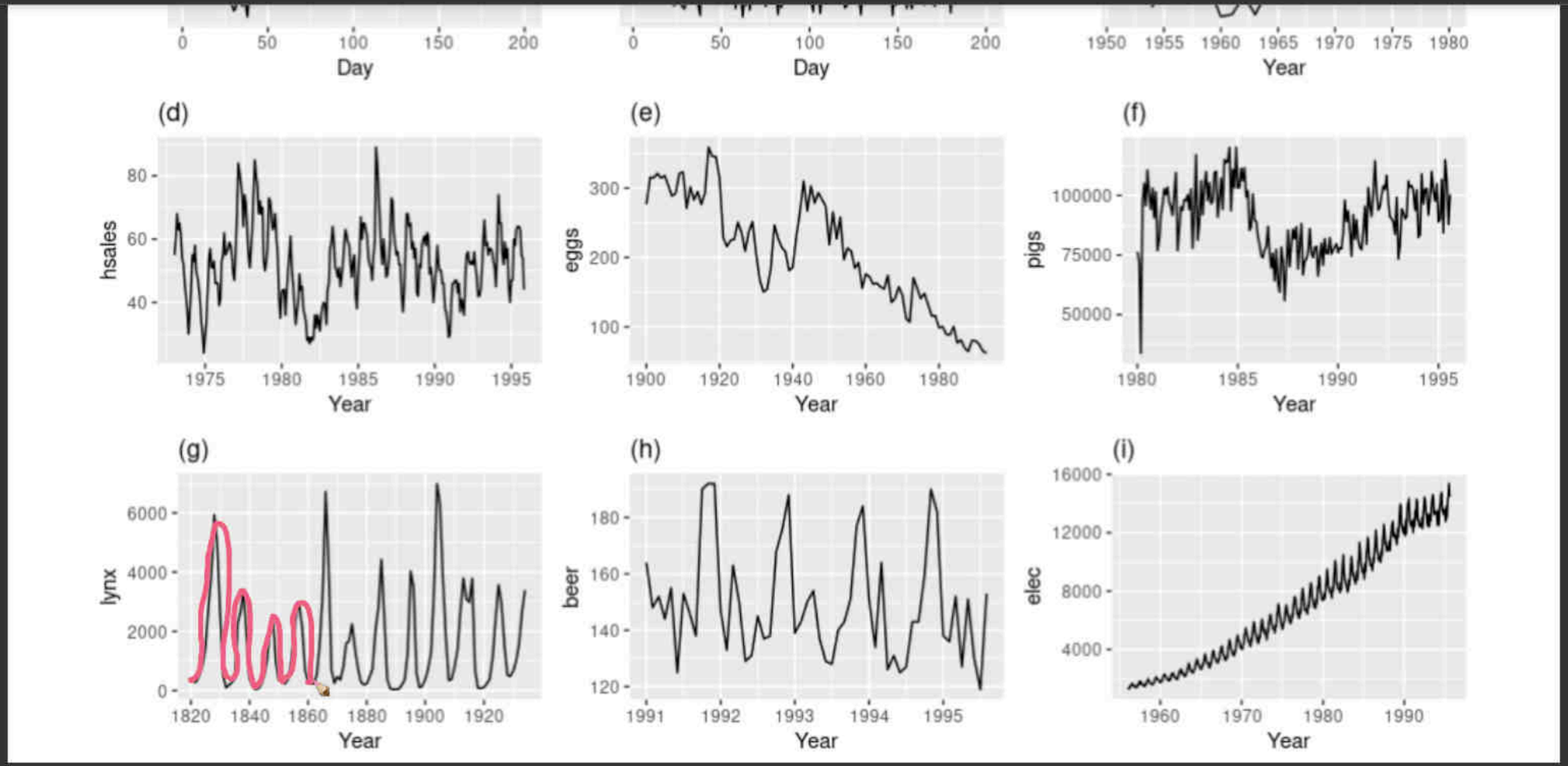
48 / 48



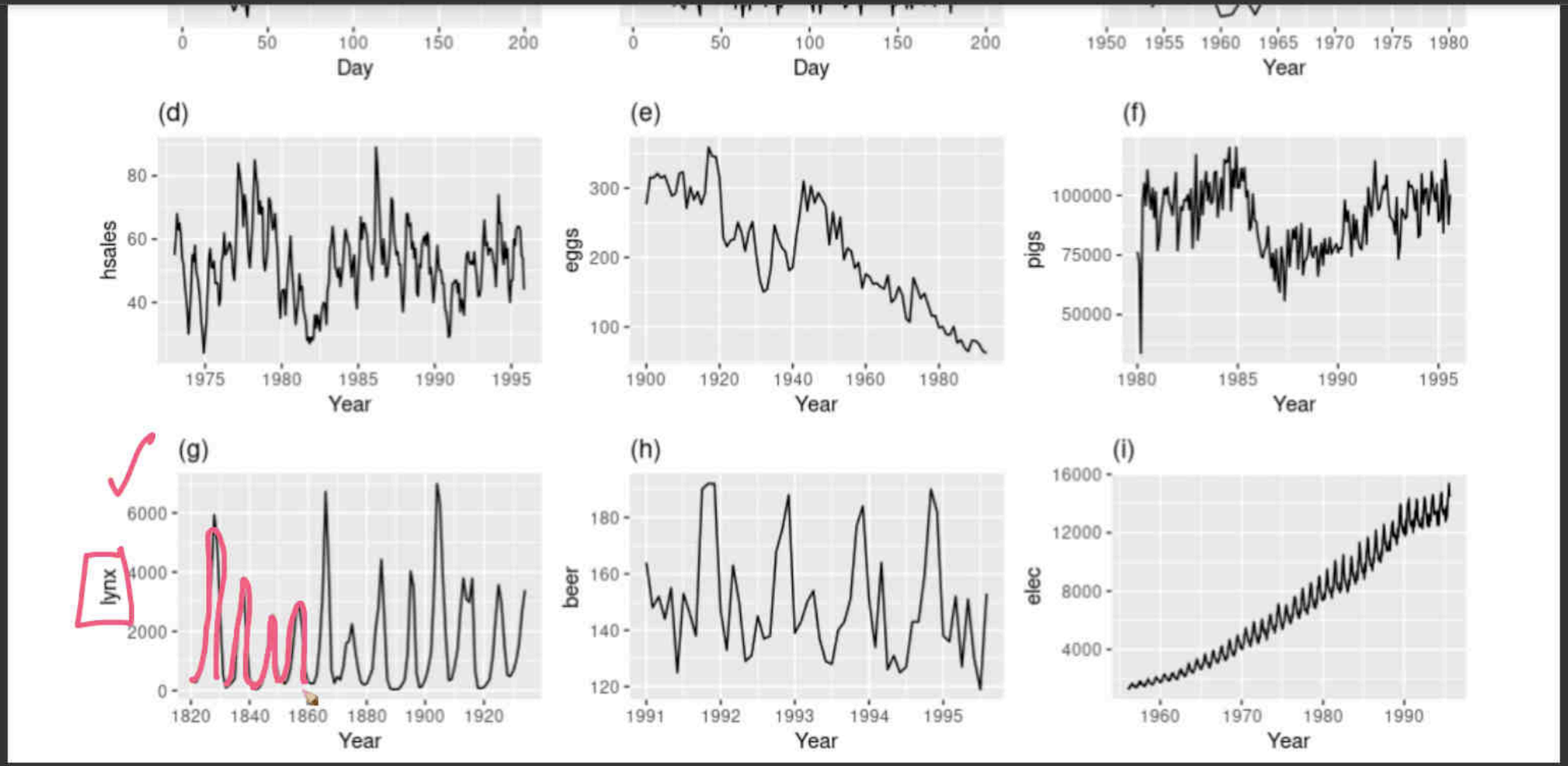
- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



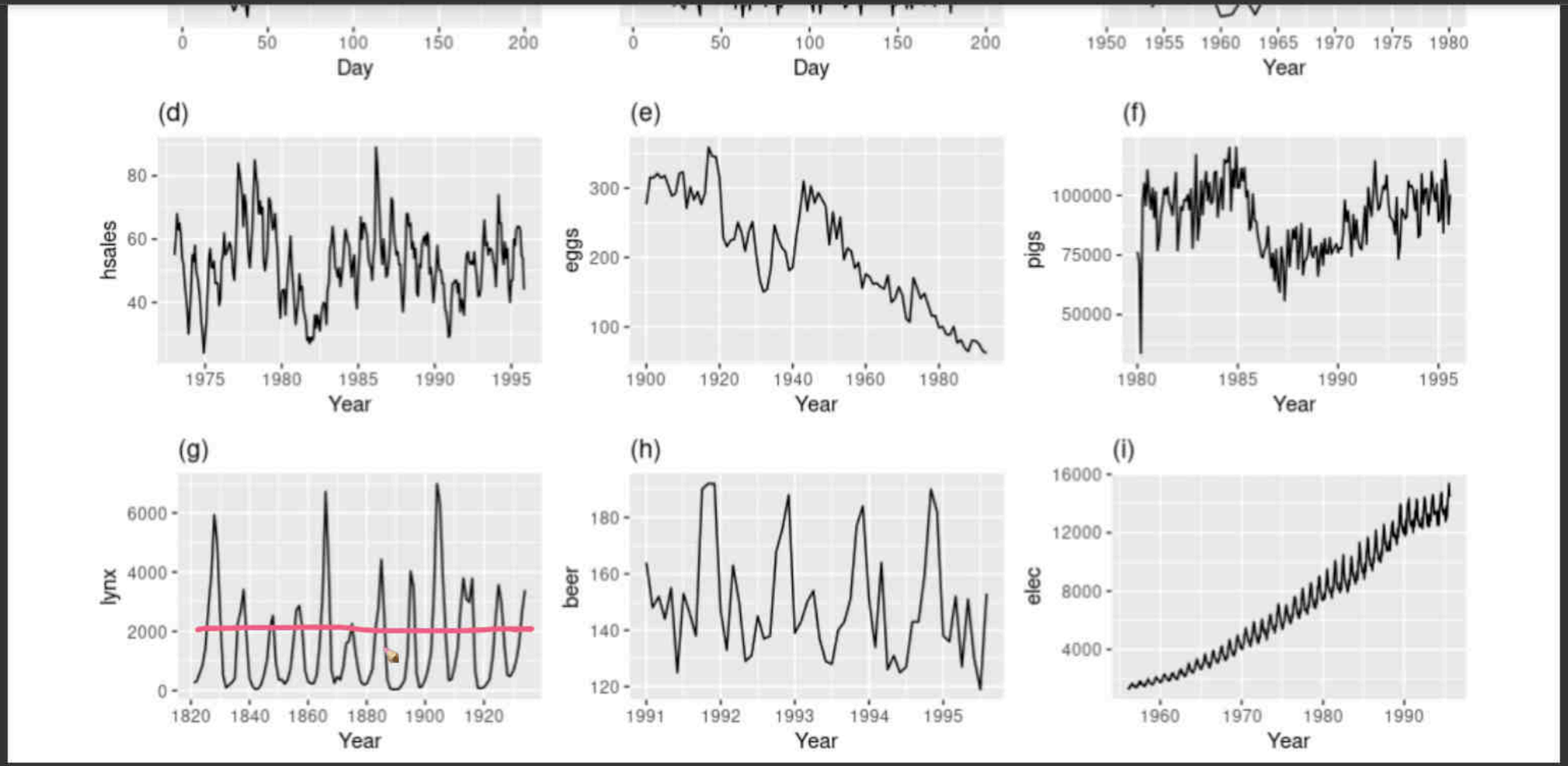
- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



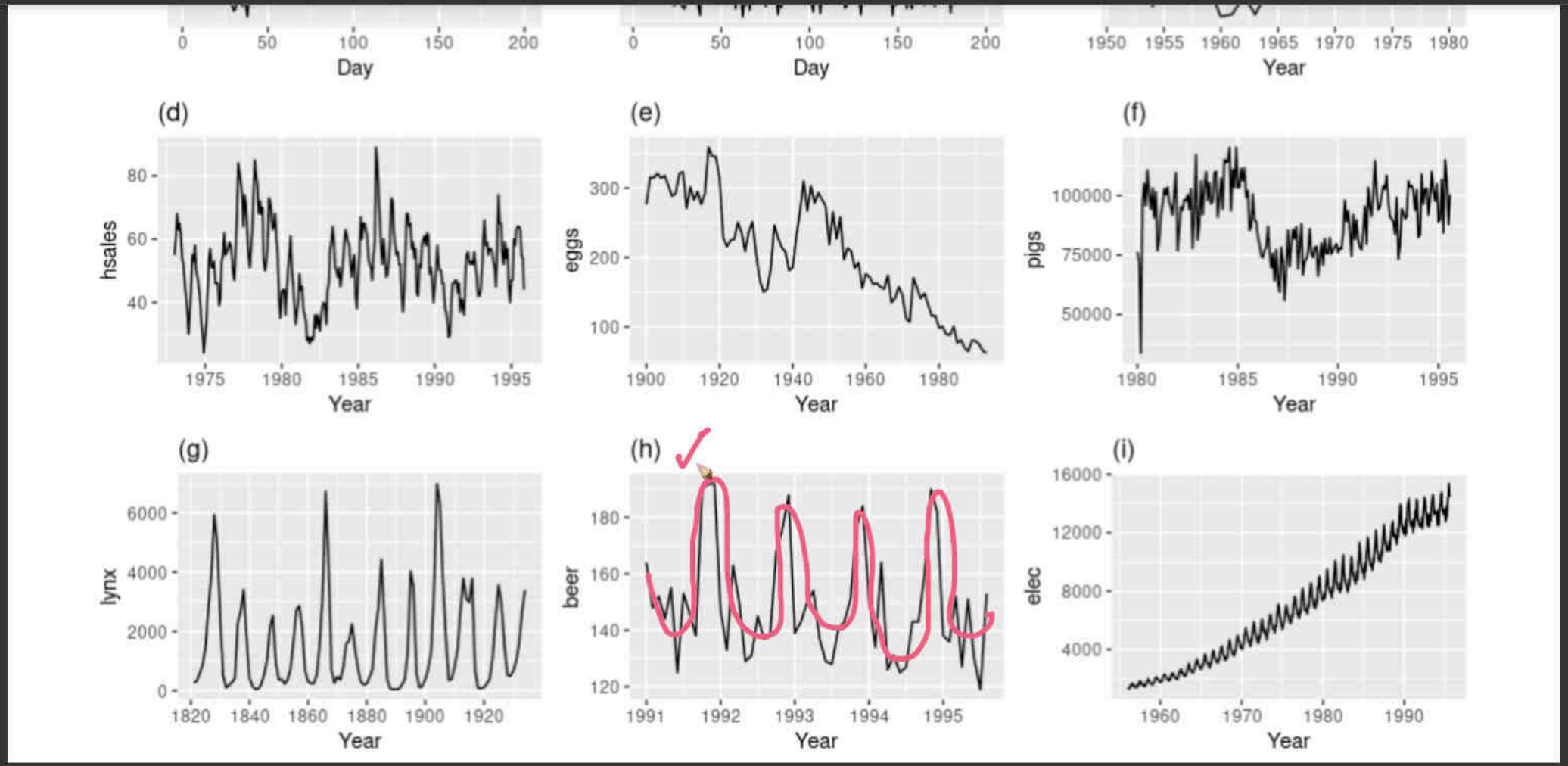
- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



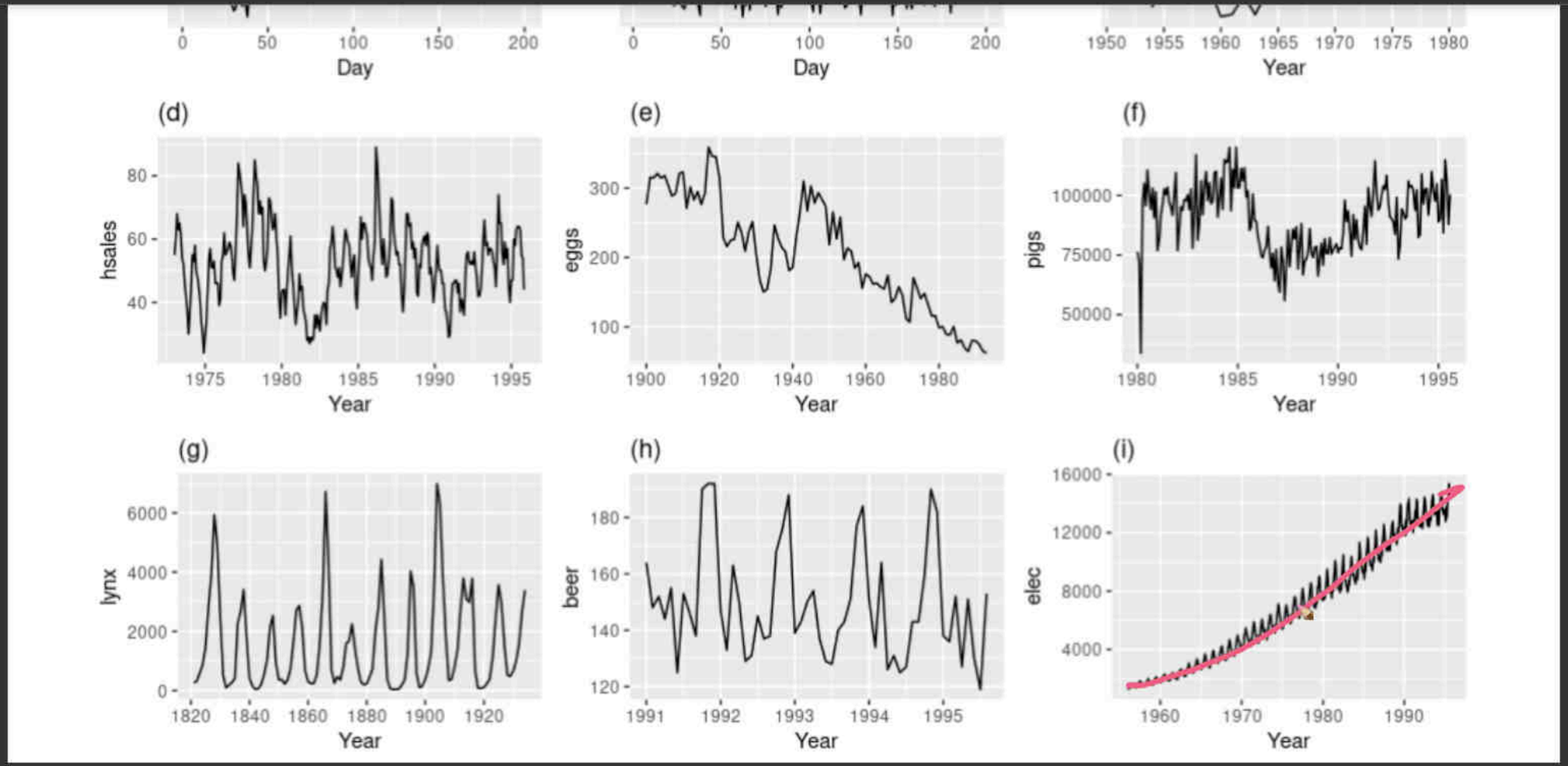
- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



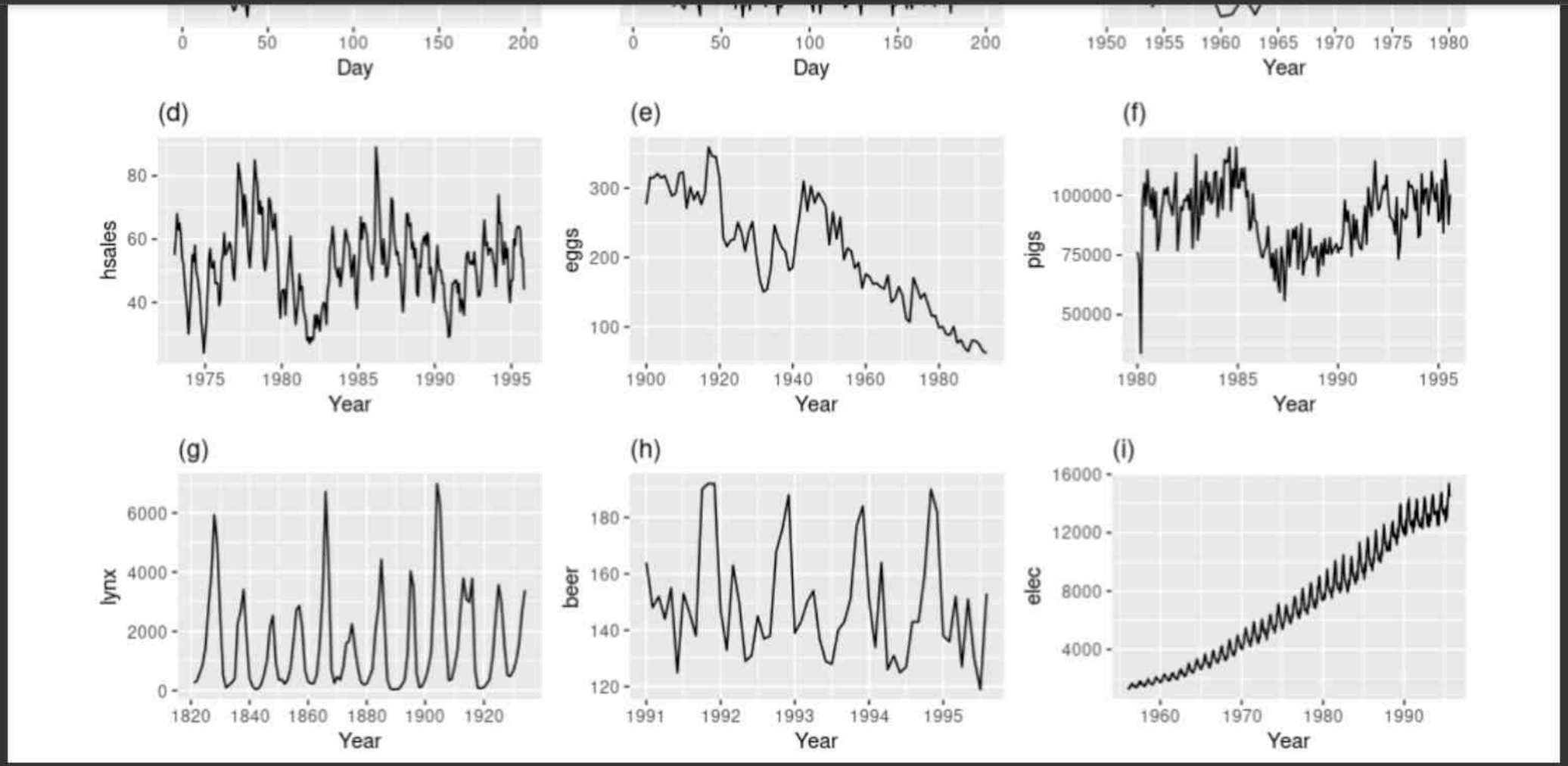
- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.



- `a, c, e, f, :` Not stationary
  - either have a trend, or
  - mean changing with time.

Visual analysis for stationarity → tricky

Test → Dickey fuller Test

p-value

AD-Fuller

Augmented DF test

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:40
Forecasting a | Time-Series | Stationary pr | pandas.DataF | Partial autoc | Exponential s | Prophet | For | Forecasting a | pandas.Data | Dickey–Fuller | statsmodels | New Tab
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=ppWFLFgpNNkW
+ Code + Text
RAM Disk

- `g:` Stationary
  - Predicting this is dicey, so we assume it to be stationary, and try building model, and seeing if it performs.
  - Looks like a cyclic time series. But these are not at regular intervals.
  - So, even though there is some seasonality, it can't be predicted.

Categorizing a time series by just looking at it can be a little subjective.

For example, somebody may argue that `plot b` is non-stationary, as it looks like there may be some seasonality there.

{ H0: data is not stat..

```
sm.tsa.stattools.adfuller(mobile_sales.Sales)[1]
```

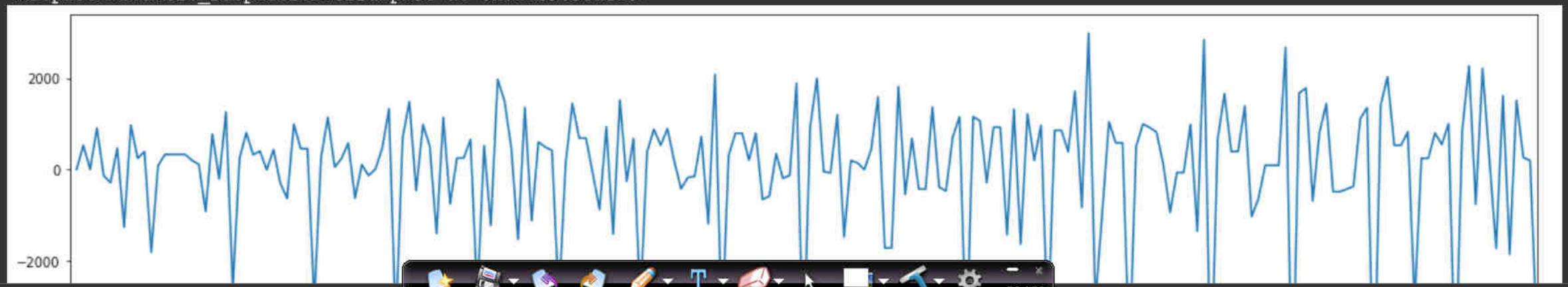
```
0.9789233920172066
```

p-value ~ 0.98 < 5% → reject H0

↳ accept H0

```
# Detrending
detrend = mobile_sales.Sales.diff()
detrend.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff09653de10>
```



58 / 59

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:45

De-Trending

remove the trend

linear: $m \times t + c$

$$x(t) = b(t) + s(t) + e(t)$$

$$x(t) = [mt + c] + s(t) + e(t)$$

$$x'(t) = \frac{d(x(t))}{d(t)} = m + \frac{ds(t)}{dt} + \frac{de(t)}{dt}$$

de Trended

59 / 61

$x(t)$ is non-stationary

$$\frac{d(x(t))}{dt} \rightarrow \lim_{\Delta t \to 0} \frac{x(t+\Delta t) - x(t)}{\Delta t}$$

$$\approx \frac{x(t+1) - x(t)}{1}$$ } differencing

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:47

differencing $\xrightarrow{\approx}$ derivative $\longrightarrow$ detrending

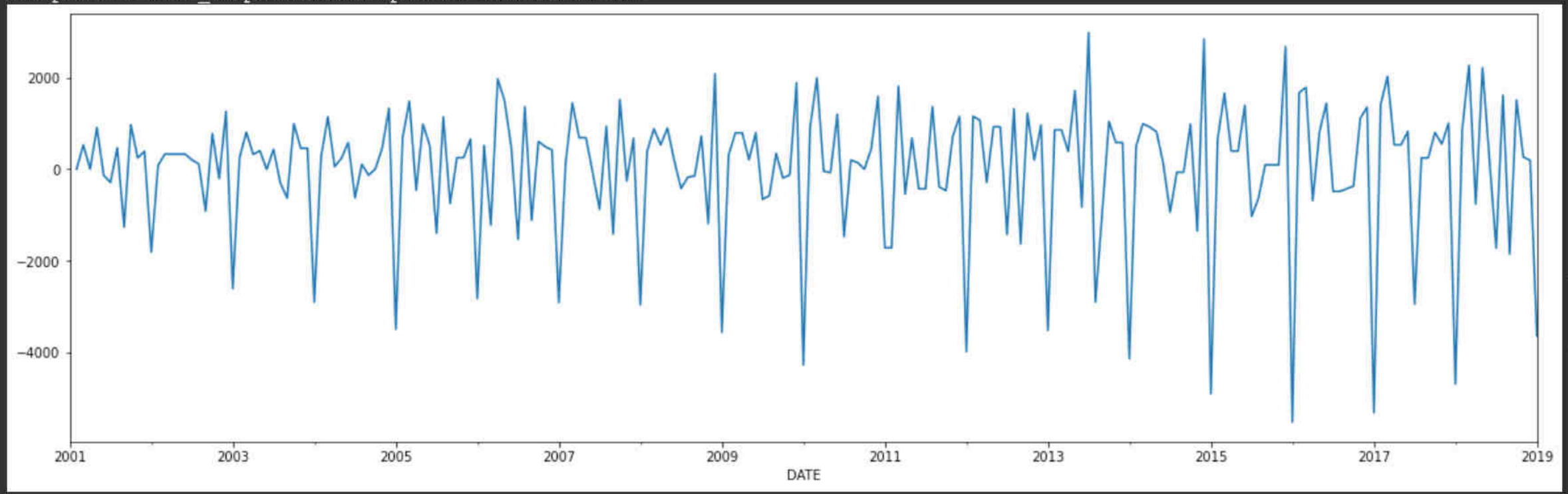
61 / 61

```
0.9789233920172066
```

```python
# Detrending
detrend = mobile_sales.Sales.diff()
detrend.plot()
```

$x(t) \rightarrow$ ADFuller $\rightarrow$ Not stat

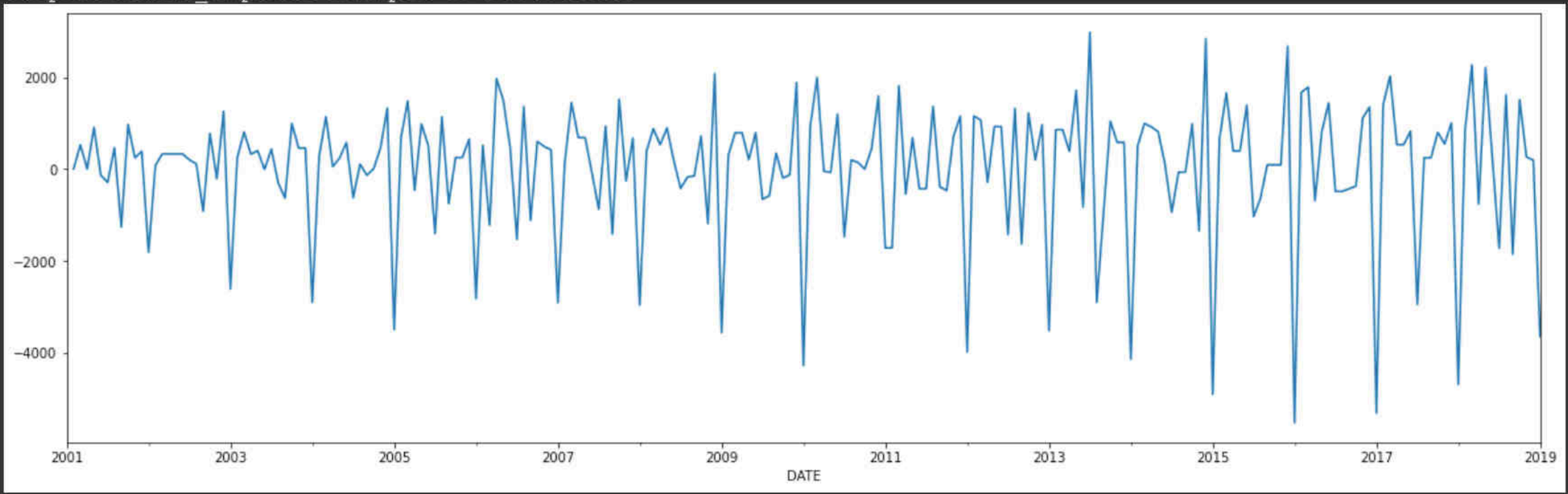
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff09653de10>
```




```python
[37] sm.tsa.stattools.adfuller(detrend.dropna())[1]
```

```
1.1781888996317722e-16
```

[34]




```
sm.tsa.stattools.adfuller(detrend.dropna())[1]
```

```
1.1781888996317722e-16
```

p-val ≃ 0 < 5%

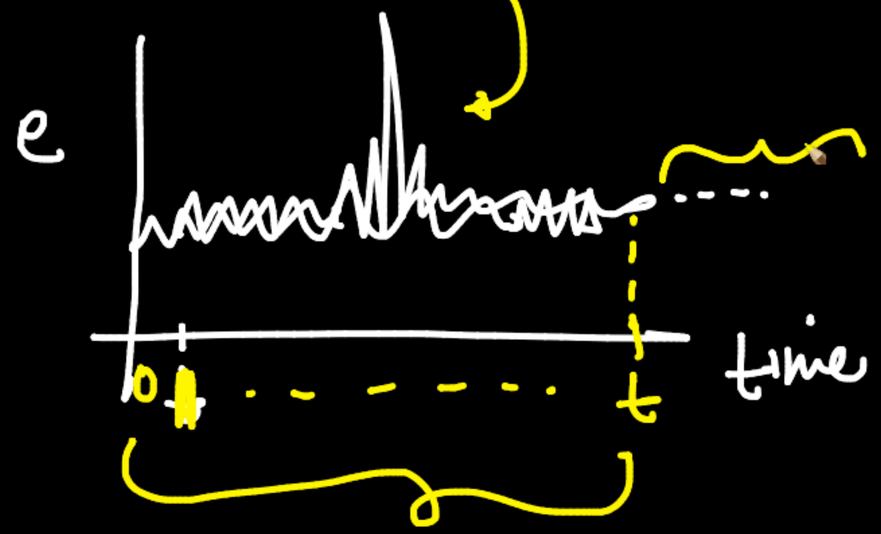
⇒ reject H_0

```
# de-seasonalisation
deseas = mobile_sales.Sales.diff(12)
deseas.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff093a804d0>
```

Q

$$x(t) = \overbrace{b(t) + s(t)}^{\hat{x}(t)} + \underbrace{e(t)}_{\text{non-predictable}}$$



Q

why de-trending

$$x(t) = b(t) + s(t) + e(t)$$

$$[x(t) - b(t)] = c + s(t) + e(t)$$

error-analysis

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:54
65 / 65

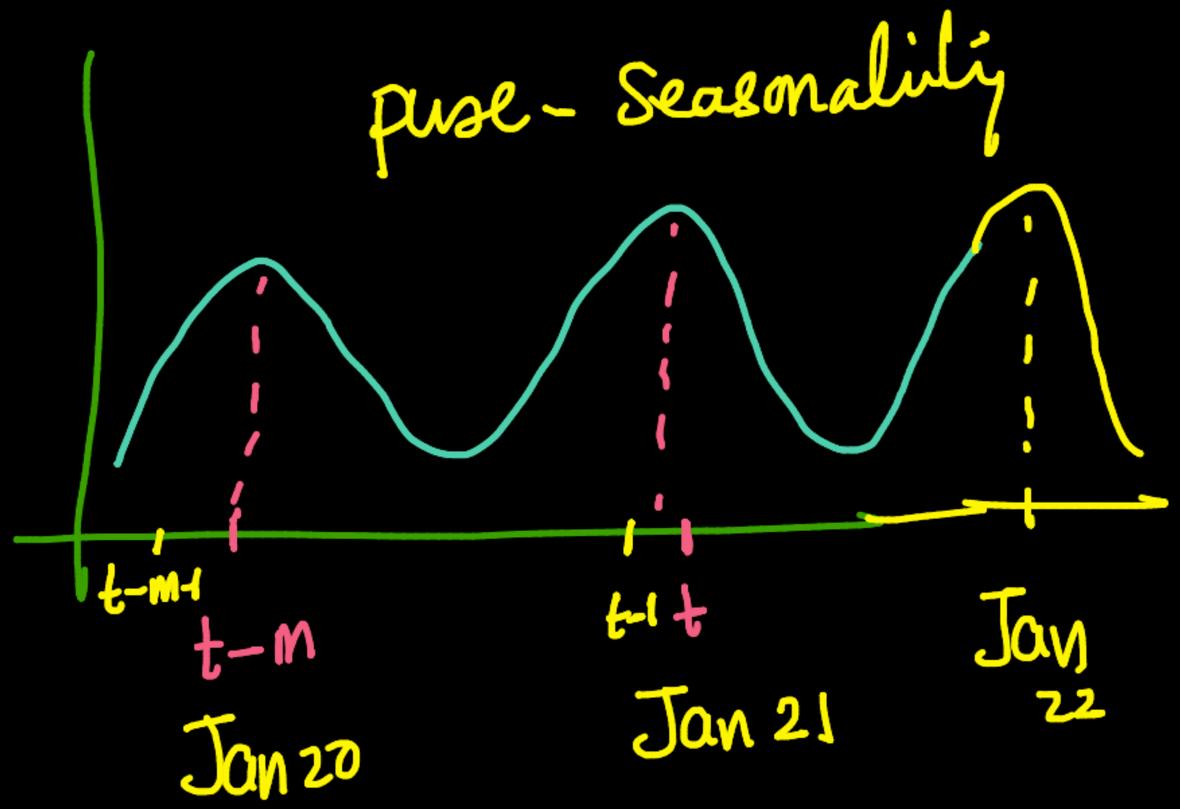
Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:57

INTUITION

Seasonality

↓

m-differencing



$$y_t \approx y_{t-m}$$

$$k = 12$$

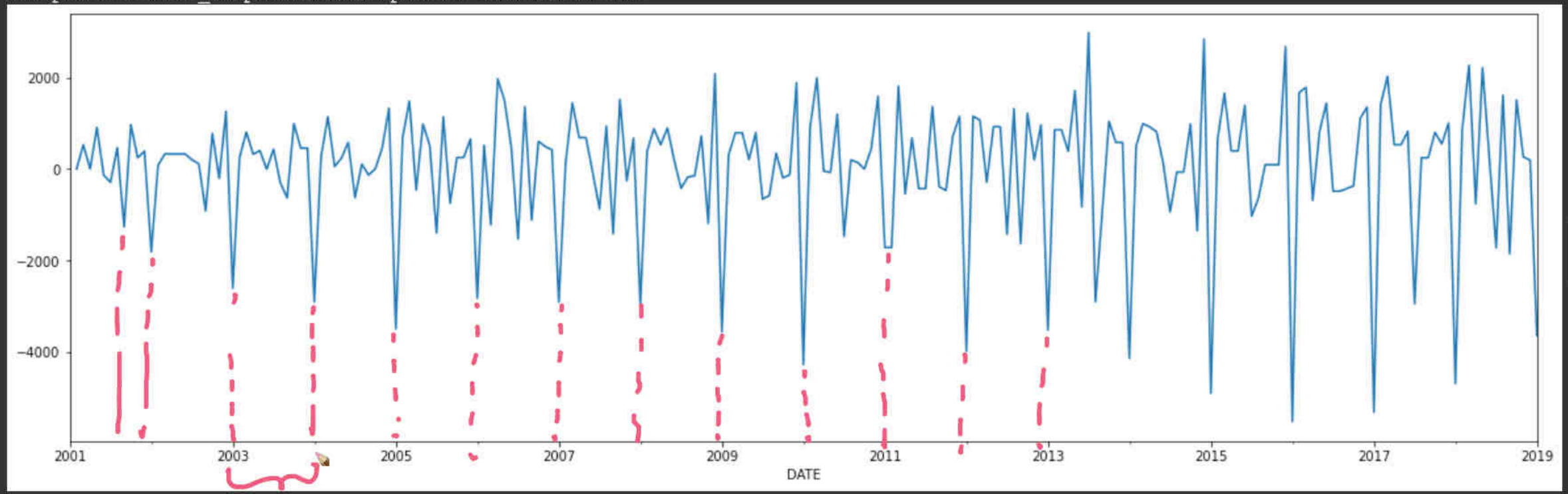
66 / 66

```
sm.tsa.stattools.adfuller(mobile_sales.Sales)[1]
```

```
0.9789233920172066
```

```
# Detrending
detrend = mobile_sales.Sales.diff()
detrend.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff09653de10>
```




```
sm.tsa.stattools.adfuller(detrend.d
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Forecasting a | Time-Series | Stationary pr | pandas.DataF | Partial autoc | Exponential s | Prophet | For | Forecasting a | pandas.Data | Dickey–Fuller | statsmodels | New Tab
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=LhjKJAIbPfnR
+ Code + Text Reconnect

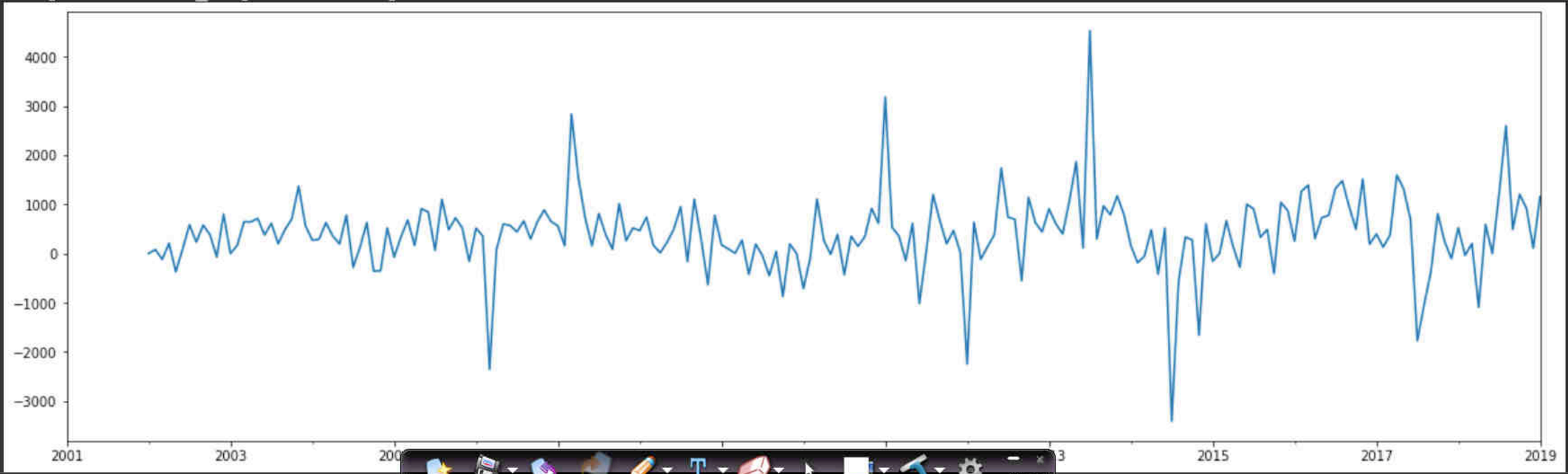
```
[ ] sm.tsa.stattools.adfuller(detrend.dropna())[1]
```

```
1.1781888996317722e-16
```

```
# de-seasonalisation
deseas = mobile_sales.Sales.diff(12)
deseas.plot()
```

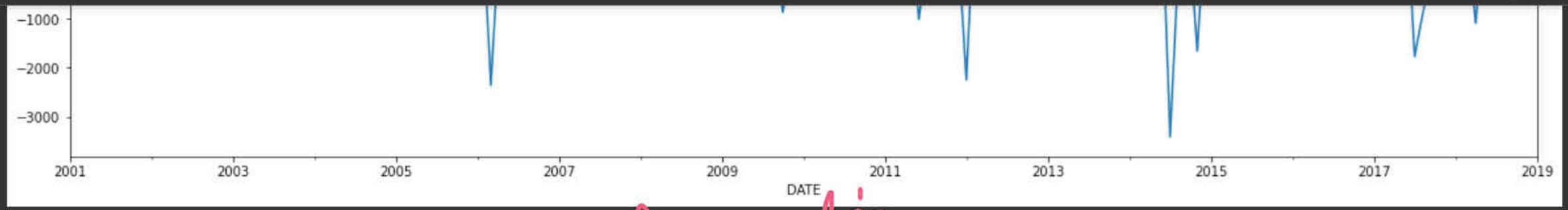
m

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff093a804d0>
```



68 / 68

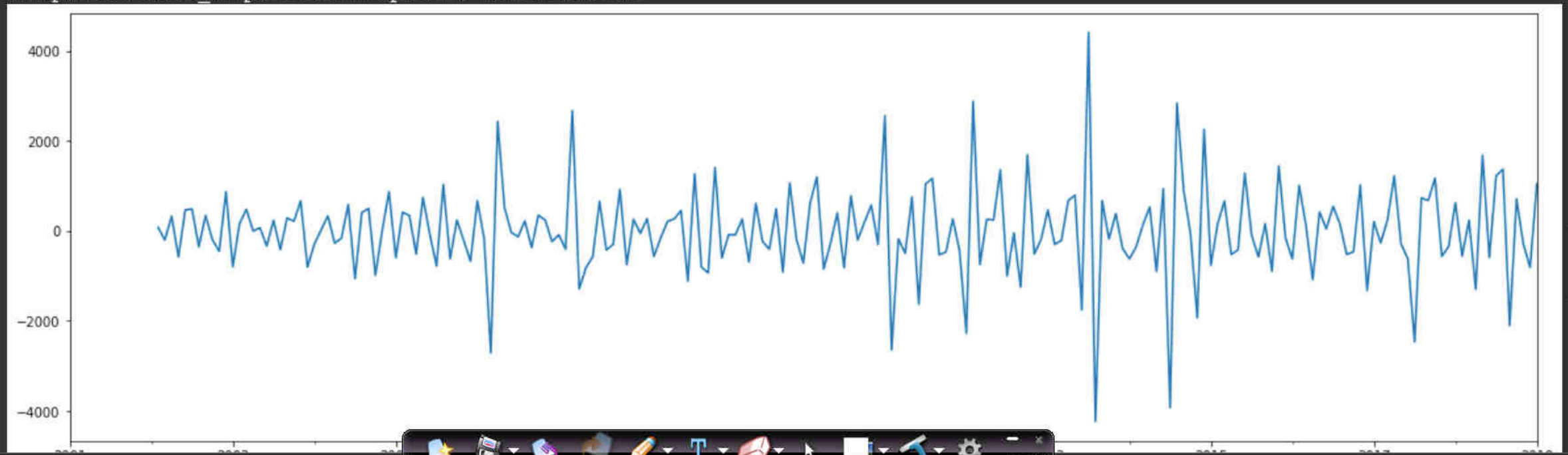
Chrome File Edit View History Bookmarks Profiles Tab Window Help 11:00
Forecasting a | Time-Series | Stationary pr | pandas.DataF | Partial autoc | Exponential s | Prophet | For | Forecasting a | pandas.Data | Dickey–Fuller | statsmodels.t | New Tab
colab.research.google.com/drive/1vLADuayVVAWtY-fo3XXbi608UUnVQPdB#scrollTo=lWOZjx_QP19a
+ Code + Text Reconnect




```
stationary = mobile_sales.Sales.diff(1).diff(12)
stationary.plot()
```

detrend → deseasonalize

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff0939ac0d0>
```



69 / 69

How to find 'm'

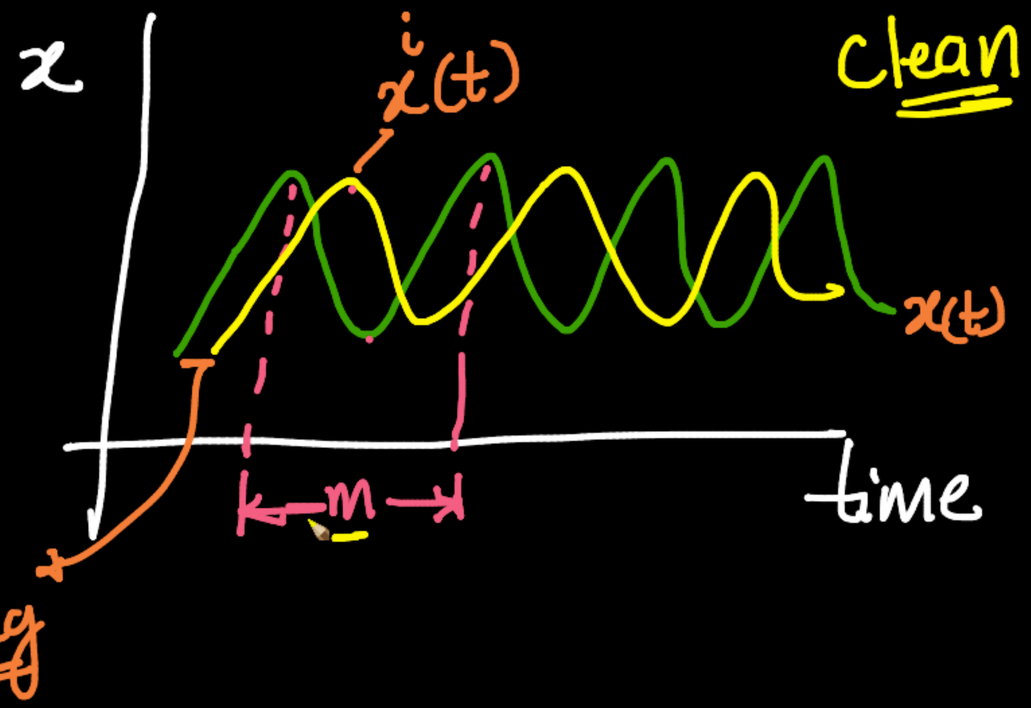
↓

Seasonality

ACF & PACF

$$\begin{cases} x(t) = b(t) + s(t) + e(t) \\ \hat{x}(t+1) = b(t+1) + s(t+1) \end{cases}$$

Q



clean

de-trended data

$i = 1, 2, 3, \ldots, m, \ldots$

x_t

when $i = m$

↓

more overlap

↳ corr. coeff

WAF to find m

– local maxima & minima & find m

– find one local min

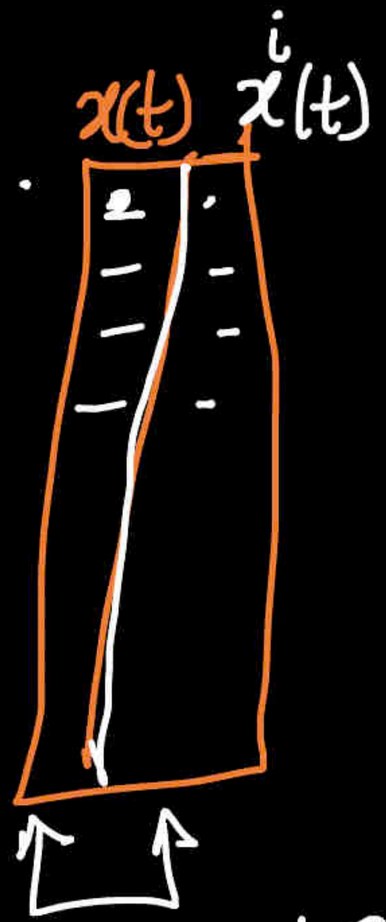
↓

next min

↓

next min

Chrome File Edit View History Bookmarks Profiles Tab Window Help



PCC ≈ 1.0

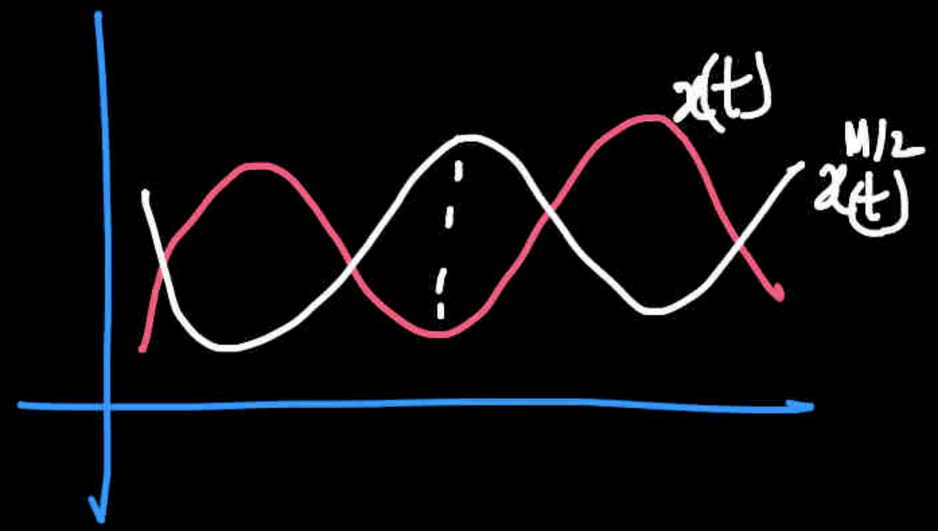
$i = m$

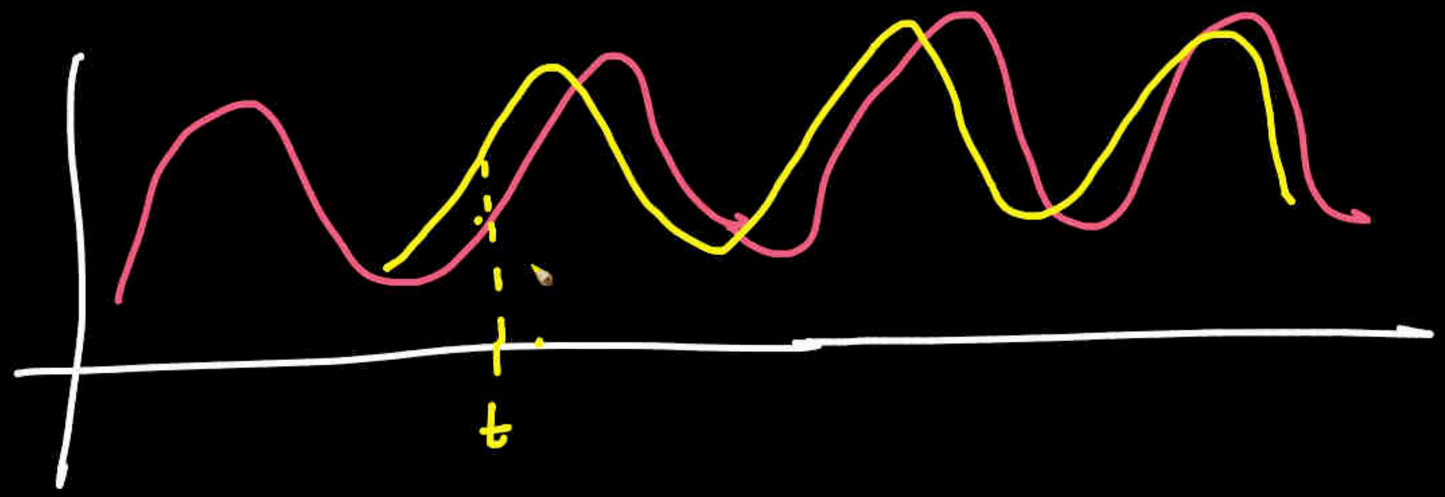
PCC → linear

-1 to 1

72 / 73

Chrome File Edit View History Bookmarks Profiles Tab Window Help



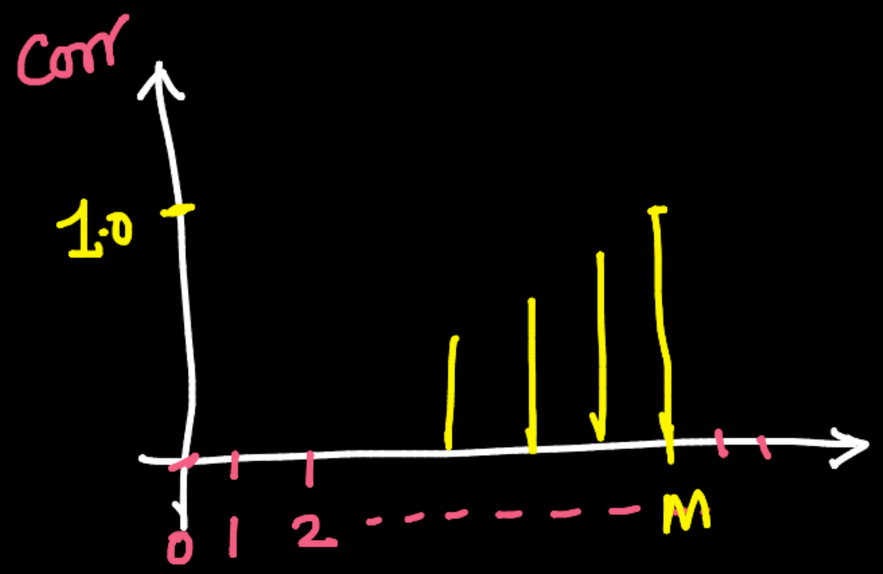


73 / 74

Chrome File Edit View History Bookmarks Profiles Tab Window Help 11:17

"lag"

Auto correlation plot



i = lags

f(lags) = corr-coeff

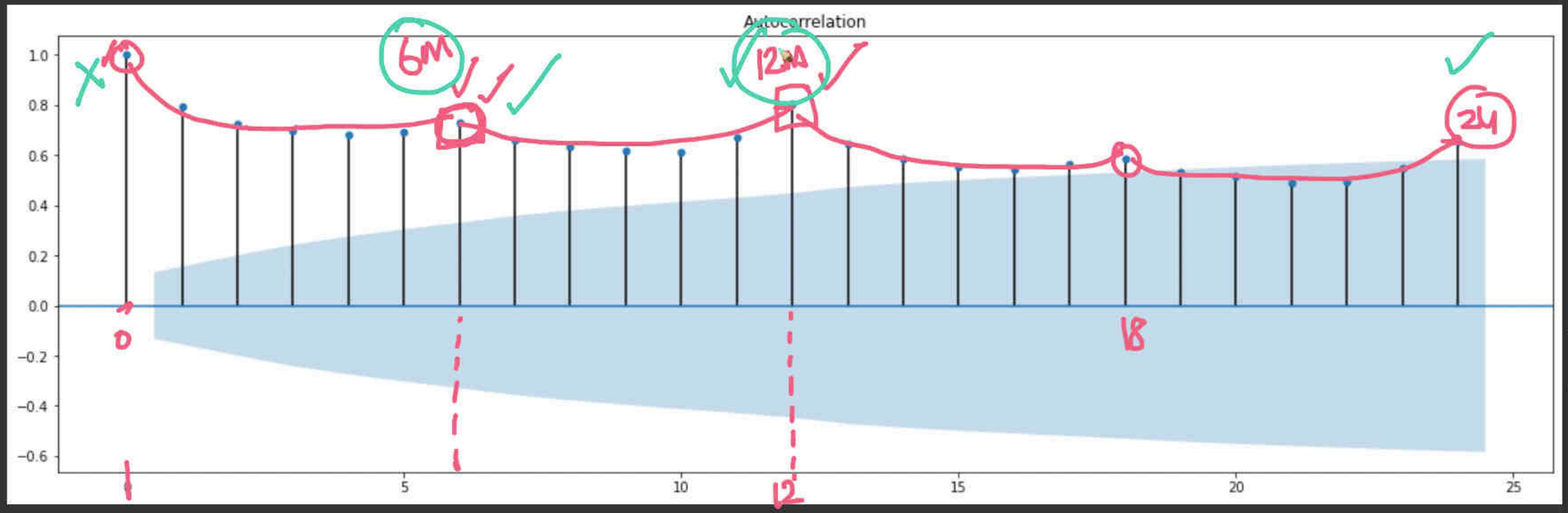
(ACF)

74 / 74

```
0.8002851477229684
```

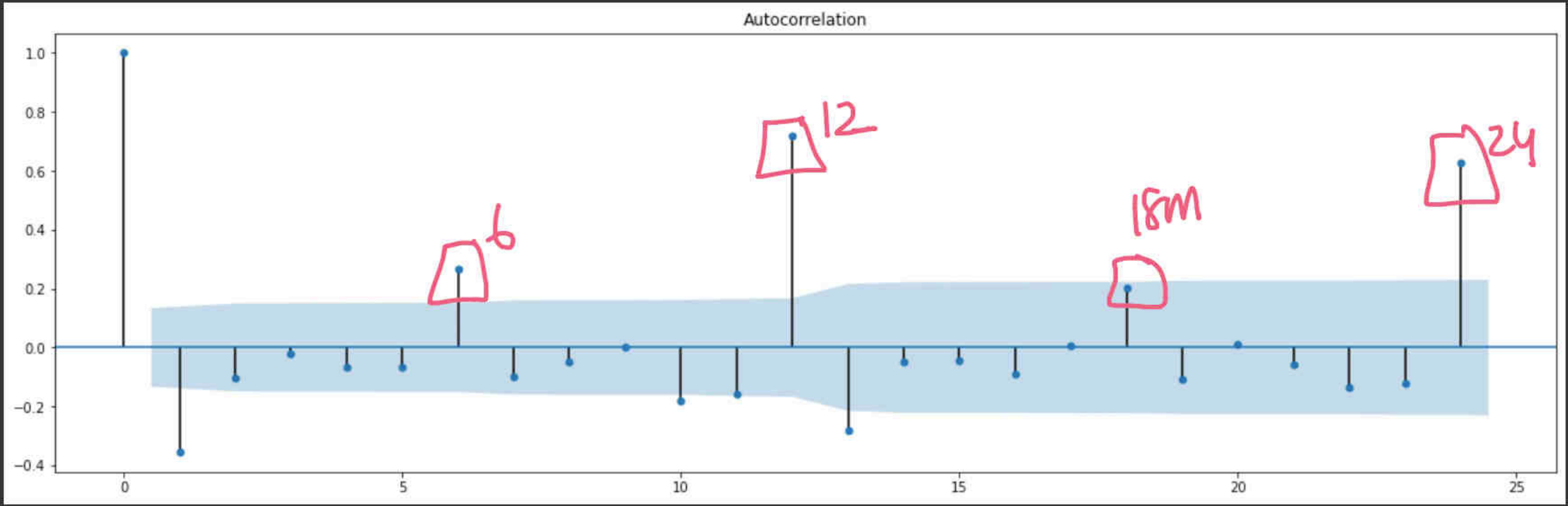
```
[ ] from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(mobile_sales.Sales);
```

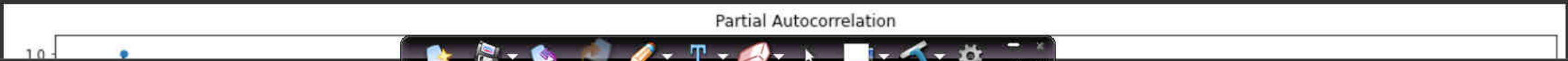



```
[ ] # for detrended time-series
```

```
# for detrended time-series
plot_acf(mobile_sales.Sales.diff().dropna());
```

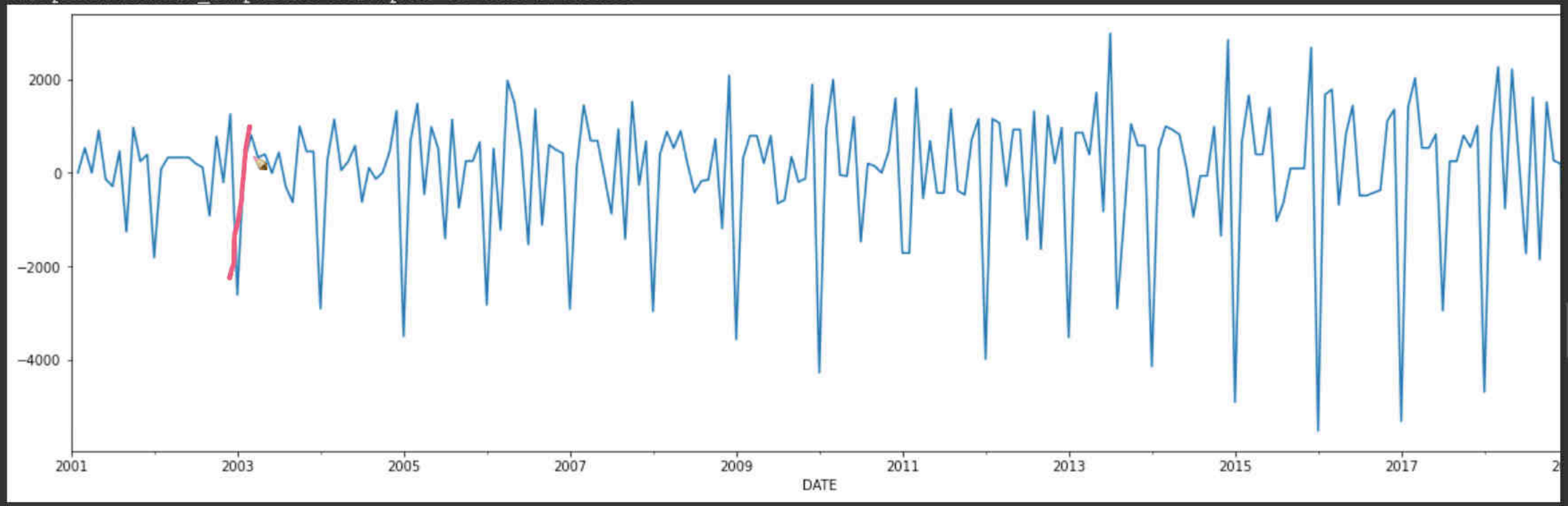



```
# PACF
plot_pacf(mobile_sales.Sales);
```



```
detrend = mobile_sales.Sales.diff()
detrend.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff09653de10>
```



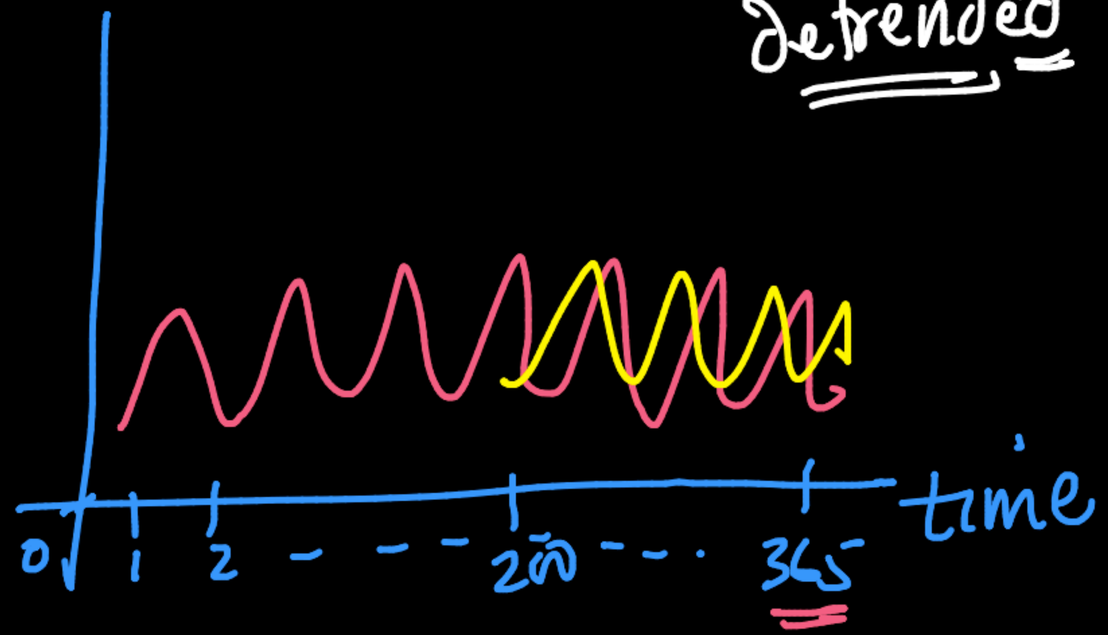

```
sm.tsa.stattools.adfuller(detrend.dropna())[1]
```

```
1.1781888996317722e-16
```

```
# de-seasonalisation
```

Q

detrended



lag : 0

⋮

30 → 0.8

⋮

60 → 0.78

⋮

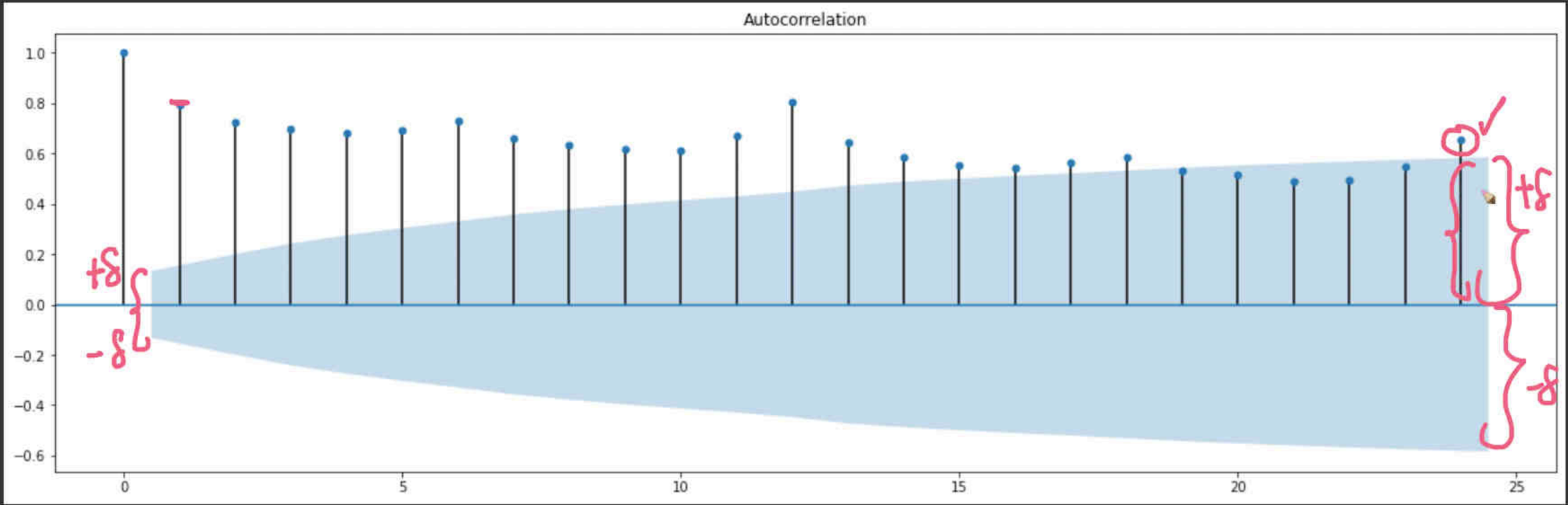
$365 - 200 = 165$

cor. coeff

20

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

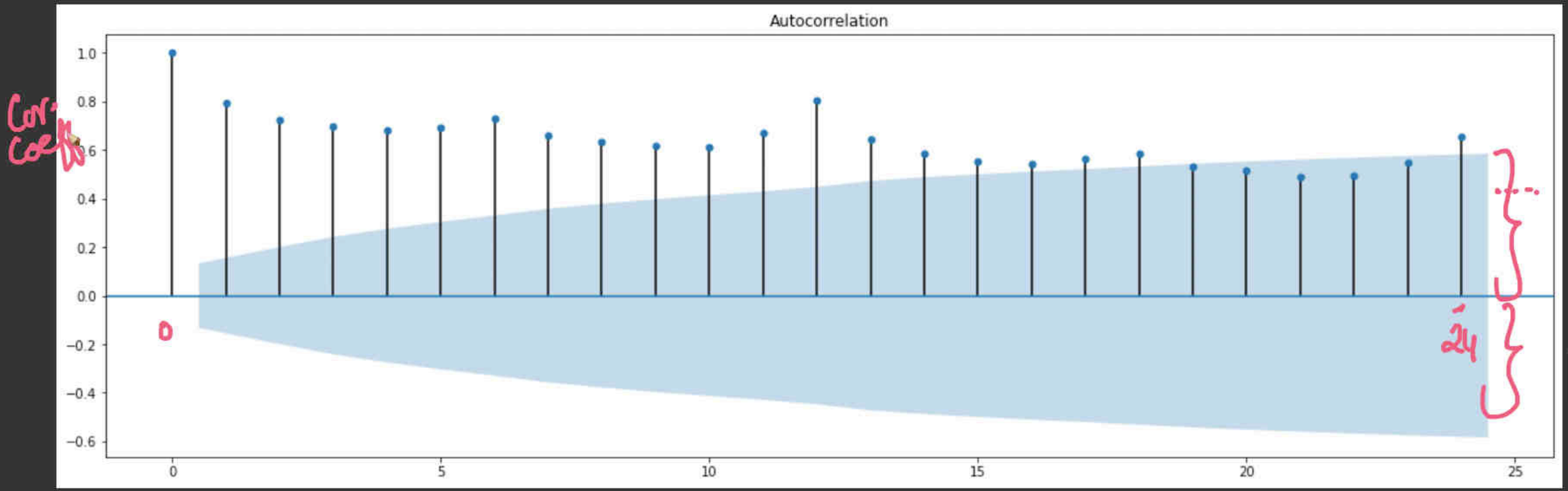
```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(mobile_sales.Sales);
```




```
# for detrended time-series
plot_acf(mobile_sales.Sales.diff().
```

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

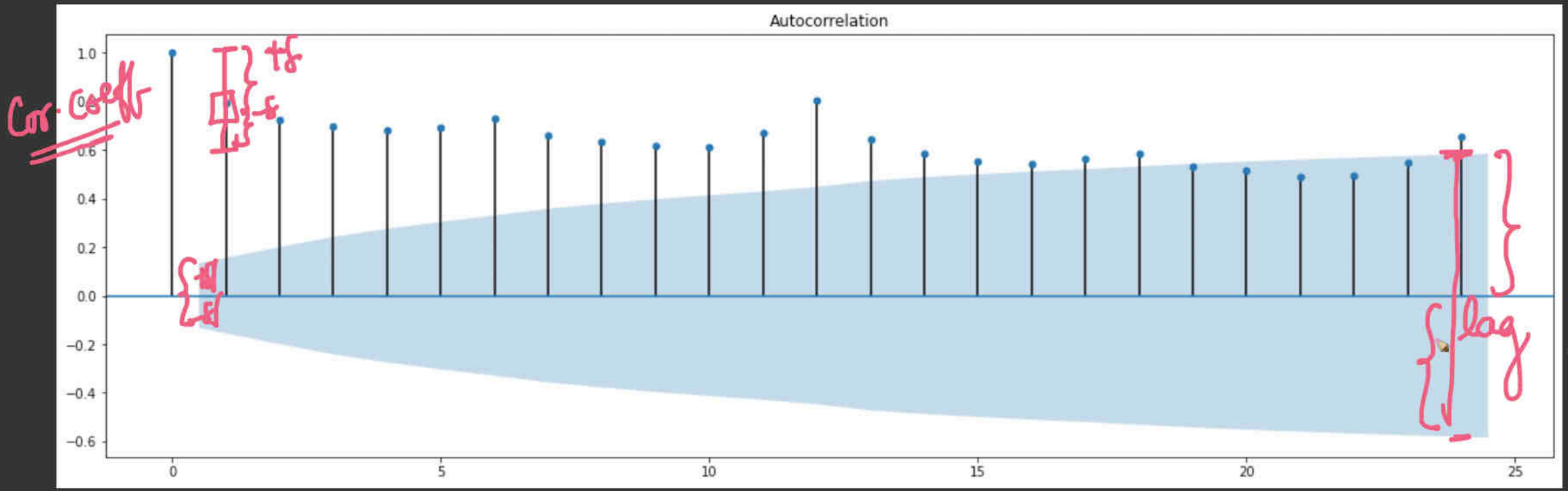
```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(mobile_sales.Sales);
```




```
# for detrended time-series
plot_acf(mobile_sales.Sales.diff().
```

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(mobile_sales.Sales);
```




```
# for detrended time-series
plot_acf(mobile_sales.Sales.diff().
```

Q

Lr-reg: x_i & x_{i-1} are indep

obs

Time-series:

$$x(t) = \underbrace{b(t)}_{\text{trend}} + \underbrace{s(t)}_{\text{seasonality}} + e(t)$$

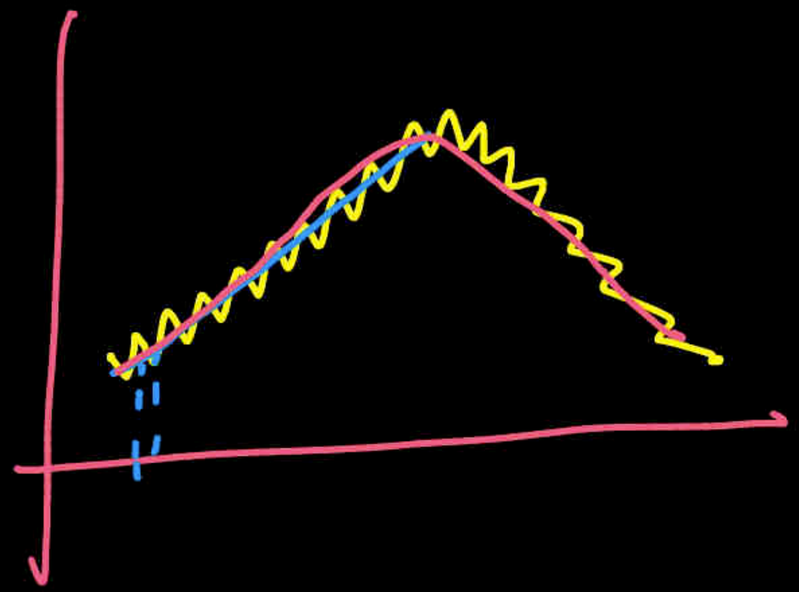
diff

tabular

$\begin{cases} mt + c \\ \text{Complex} \ldots \text{MA} \end{cases}$

e_t, e_{t-1} are indep

Chrome File Edit View History Bookmarks Profiles Tab Window Help

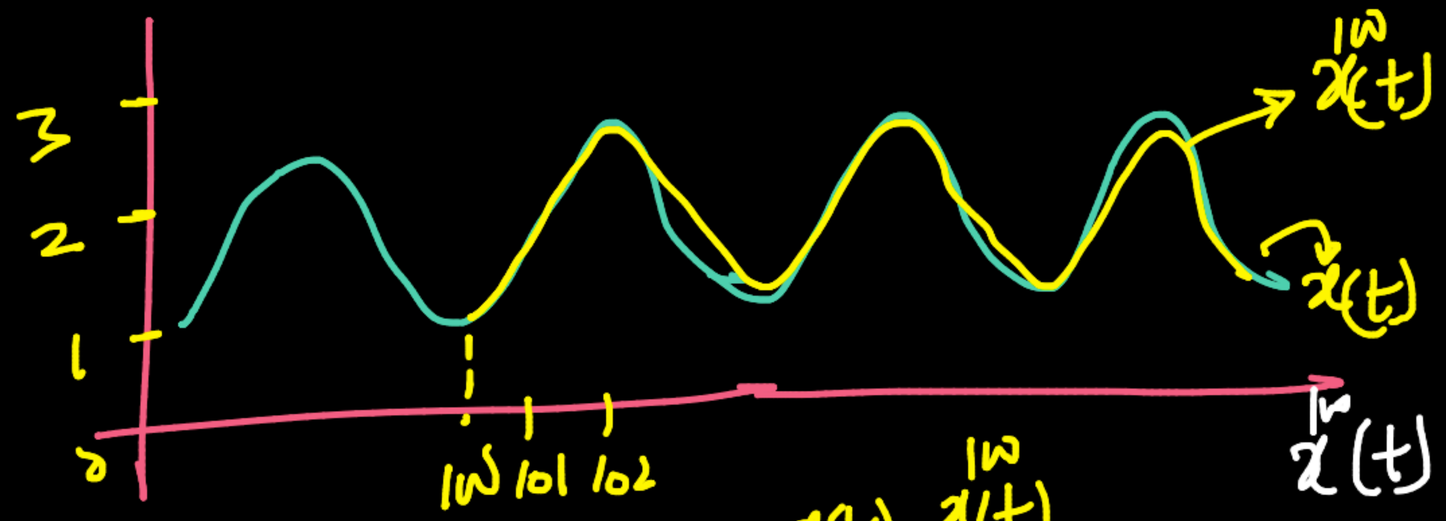


$b(t)$ → trend

$b(t)$ → diff

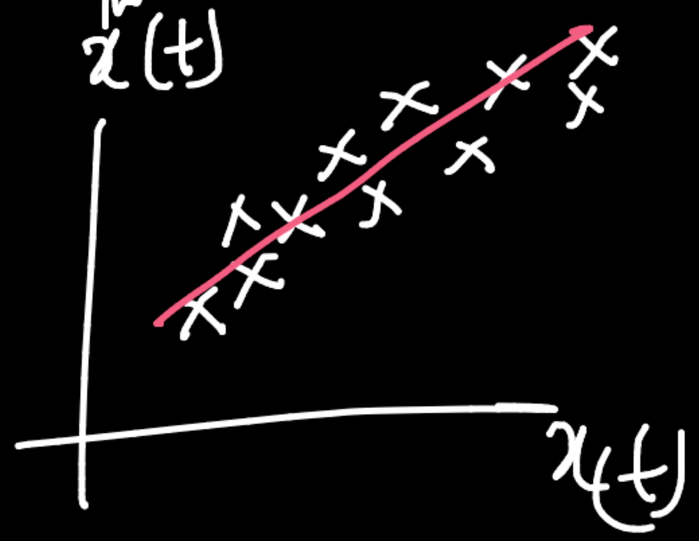
83 / 83

Q



x(t)	100 x(t)
1	1.1
1.2	1.24
⋮	⋮

PCG



Q

$\boxed{\text{RecSys}}$

$$\min \sum_{i,j} \Big[\underbrace{r_{ij}}_{\text{known}} - \big(\overbrace{\mu + b_i + b_j + u_i^T m_j}^{\text{param}} \big) \Big]^2$$

i → users, j → items

u_i, m_j

b_i, b_j

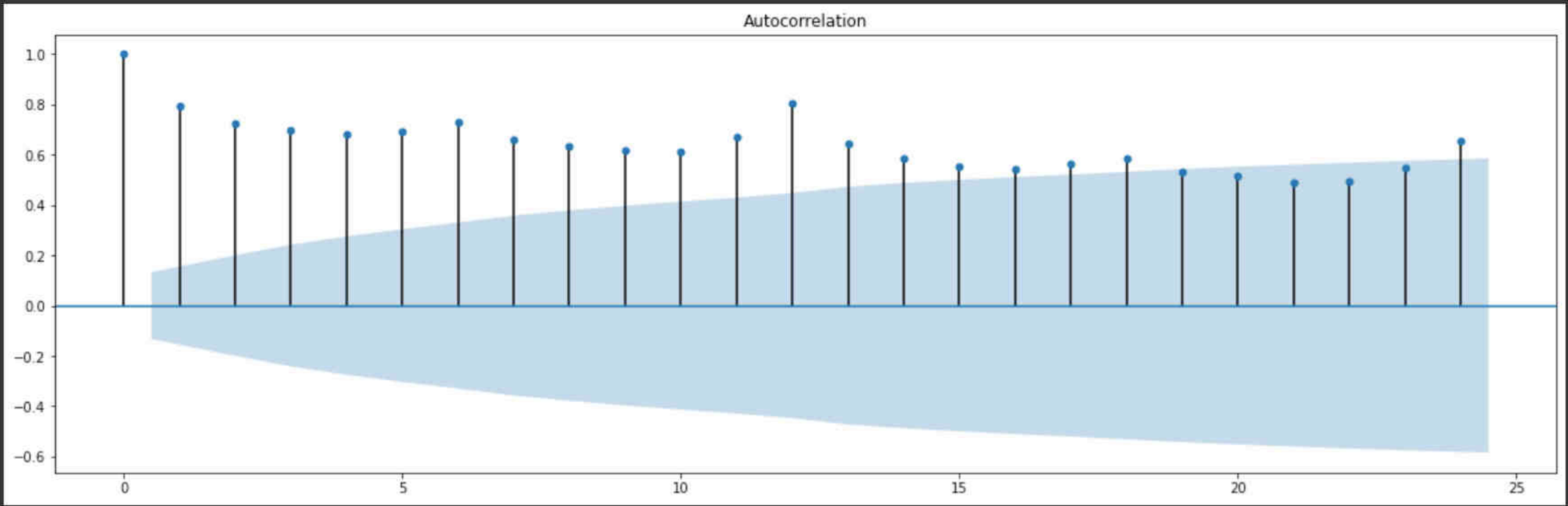
μ

$$+ \lambda_1 (b_i + b_j) +$$

$$+ \lambda_2 (\|u_i\|^2 + \|m_j\|^2)$$

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(mobile_sales.Sales);
```




```
# for detrended time-series
plot_acf(mobile_sales.Sales.diff().
```