Autocorrelation and Partial autocorrelation

VISUALIZING TIME SERIES DATA IN PYTHON

Thomas Vincent
Head of Data Science, Getty Images

Autocorrelation in time series data

- Autocorrelation is measured as the correlation between a time series and a delayed copy of itself

- For example, an autocorrelation of order 3 returns the correlation between a time series at points (`t_1` , `t_2` , `t_3` , ...) and its own values lagged by 3 time points, i.e. (`t_4` , `t_5` , `t_6` , ...)

- It is used to find repetitive patterns or periodic signal in time series

In the field of time series analysis, autocorrelation refers to the correlation of a time series with a lagged version of itself. For example, an autocorrelation of order 3 returns the correlation between a time series and its own values lagged by 3 time points.

Statsmodels

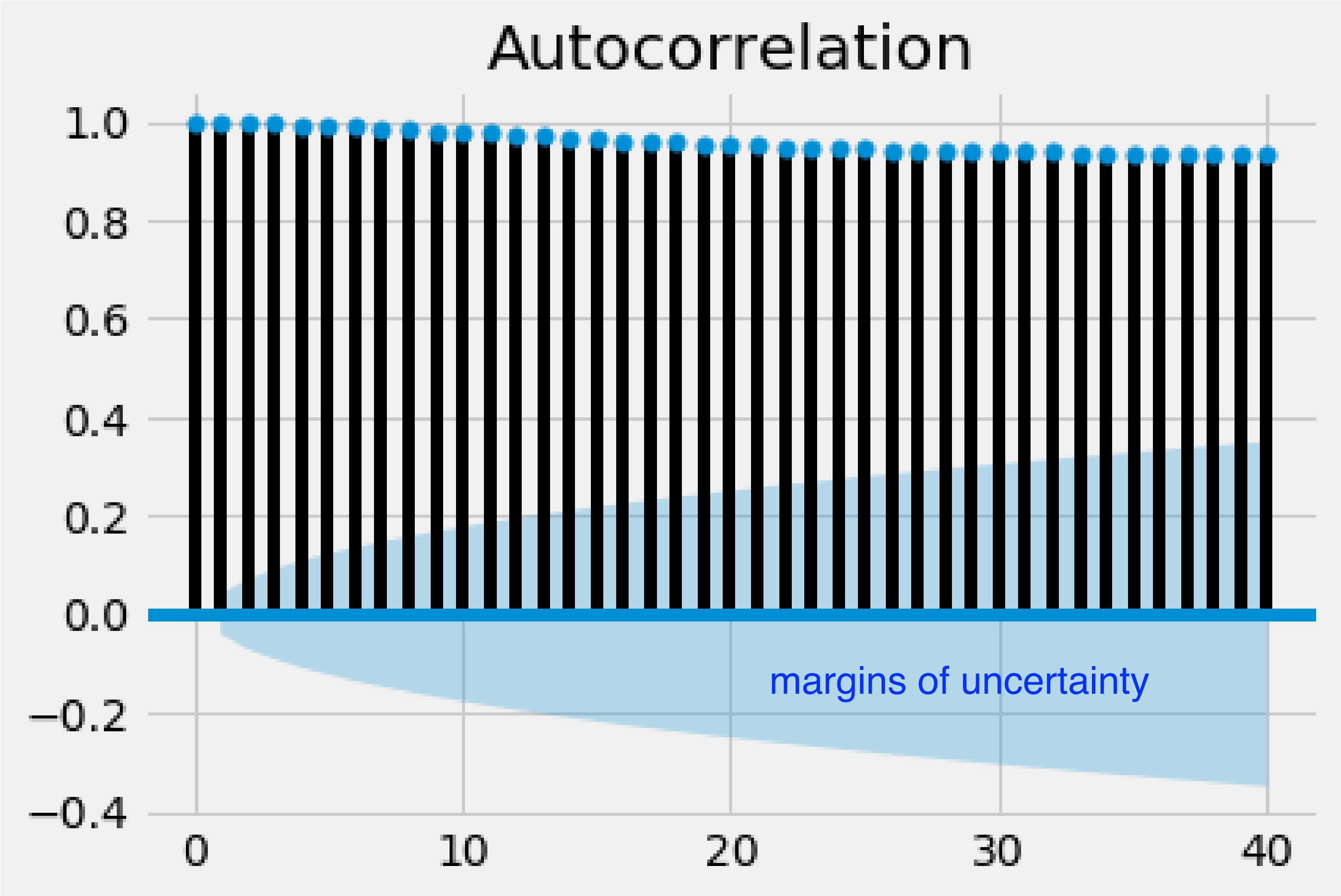
> `statsmodels` is a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration.

Plotting autocorrelations

```
import matplotlib.pyplot as plt
from statsmodels.graphics import tsaplots
fig = tsaplots.plot_acf(co2_levels['co2'], lags=40)


plt.show()
```

Interpreting autocorrelation plots



If autocorrelation values are close to 0, then values between consecutive observations are not correlated with one another. Inversely, autocorrelations values close to 1 or -1 indicate that there exists strong positive or negative correlations between consecutive observations, respectively.

In order to help you asses how trustworthy these autocorrelation values are, the plot_acf() function also returns confidence intervals (represented as blue shaded regions). If an autocorrelation value goes beyond the confidence interval region, you can assume that the observed autocorrelation value is statistically significant.

Partial autocorrelation in time series data

- Contrary to autocorrelation, partial autocorrelation removes the effect of previous time points
- For example, a partial autocorrelation function of `order 3` returns the correlation between our time series (`t1` , `t2` , `t3` , ...) and lagged values of itself by 3 time points (`t4` , `t5` , `t6` , ...), but only after removing all effects attributable to lags 1 and 2

Plotting partial autocorrelations

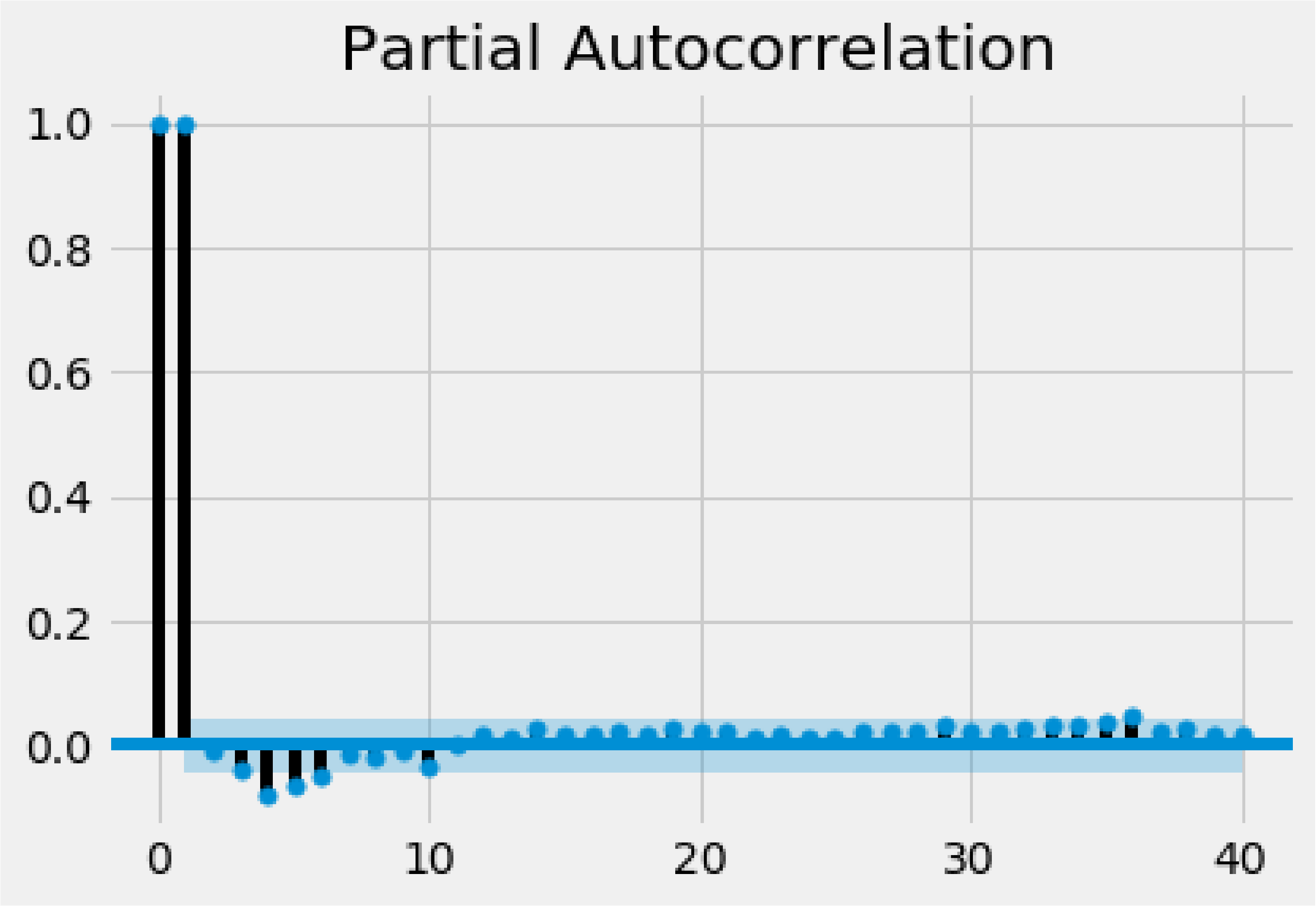
```
import matplotlib.pyplot as plt

from statsmodels.graphics import tsaplots
fig = tsaplots.plot_pacf(co2_levels['co2'], lags=40)

plt.show()
```

Like autocorrelation, the partial autocorrelation function (PACF) measures the correlation coefficient between a time-series and lagged versions of itself. However, it extends upon this idea by also removing the effect of previous time points.
For example, a partial autocorrelation function of order 3 returns the correlation between our time series (t_1, t_2, t_3, …) and its own values lagged by 3 time points (t_4, t_5, t_6, …), but only after removing all effects attributable to lags 1 and 2.

Interpreting partial autocorrelations plot



If partial autocorrelation values are close to 0, then values between observations and lagged observations are not correlated with one another. Inversely, partial autocorrelations with values close to 1 or -1 indicate that there exists strong positive or negative correlations between the lagged observations of the time series.

The .plot_pacf() function also returns confidence intervals, which are represented as blue shaded regions. If partial autocorrelation values are beyond this confidence interval regions, then you can assume that the observed partial autocorrelation values are statistically significant.

Let's practice!

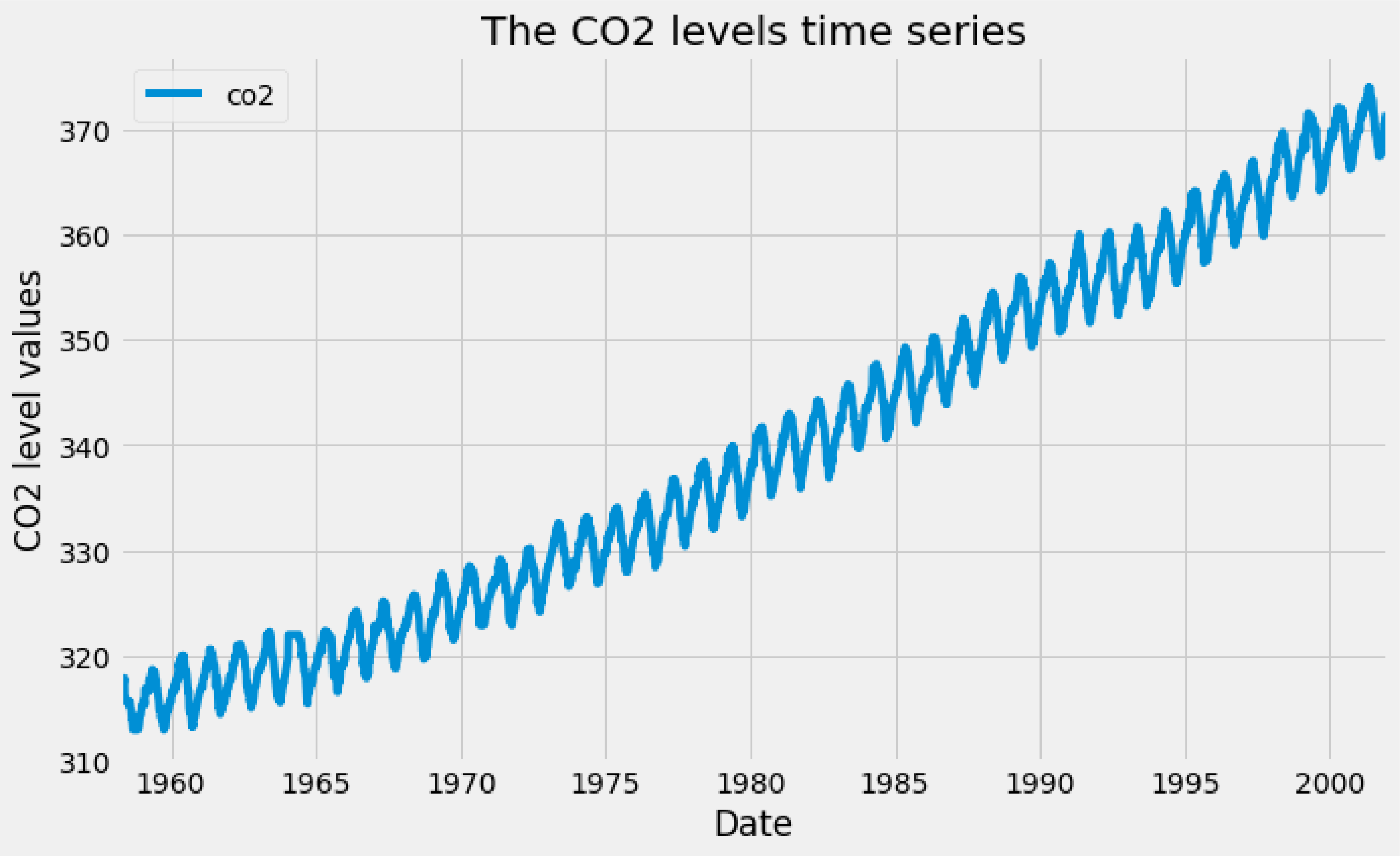
VISUALIZING TIME SERIES DATA IN PYTHON

Seasonality, trend and noise in time series data

VISUALIZING TIME SERIES DATA IN PYTHON

Thomas Vincent
Head of Data Science, Getty Images

Properties of time series



The properties of time series

- Seasonality: does the data display a clear periodic pattern?
- Trend: does the data follow a consistent upwards or downwards slope?
- Noise: are there any outlier points or missing values that are not consistent with the rest of the data?

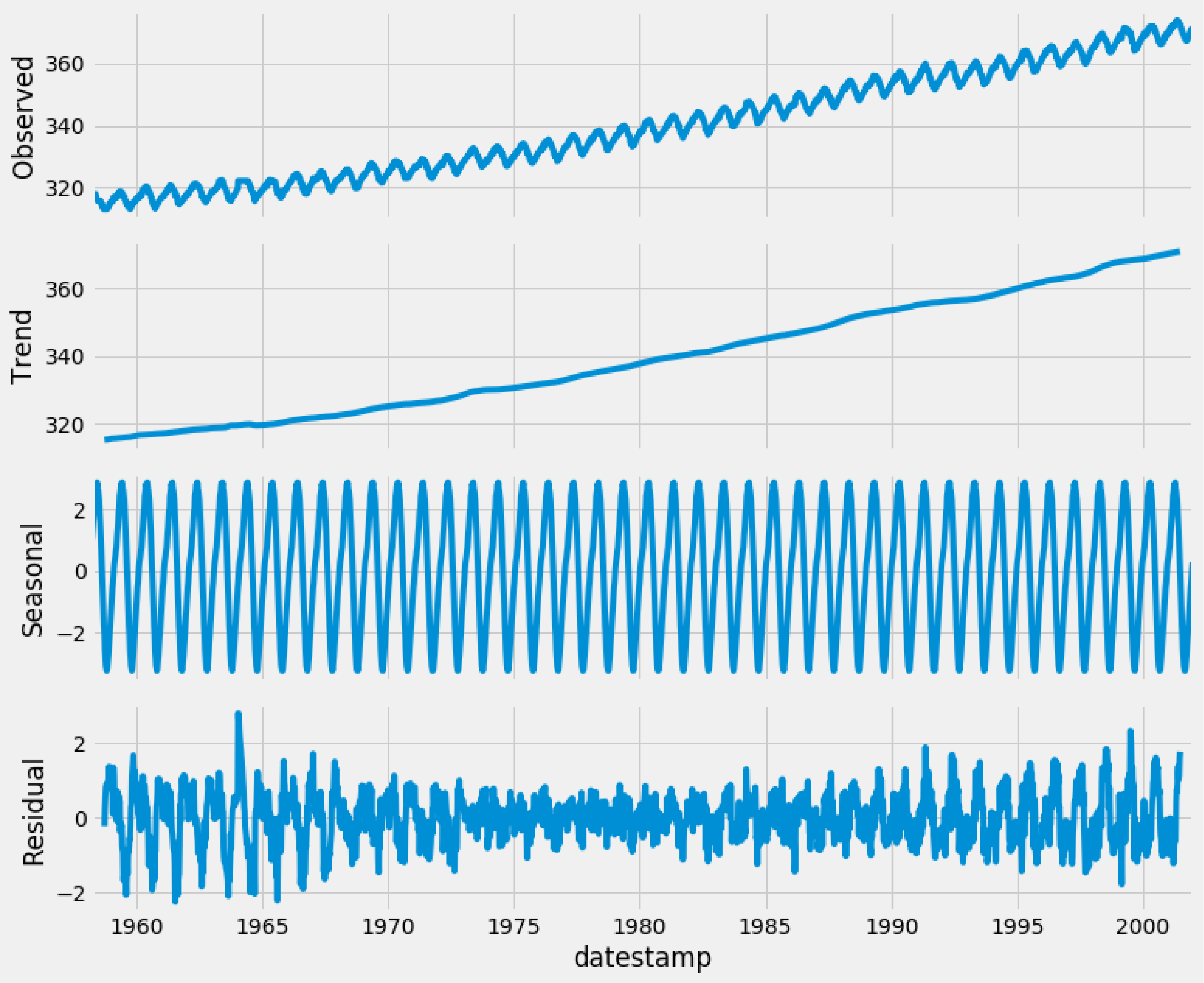
Time series decomposition

```
import statsmodels.api as sm
import matplotlib.pyplot as plt
from pylab import rcParams

rcParams['figure.figsize'] = 11, 9
decomposition = sm.tsa.seasonal_decompose(
                co2_levels['co2'])
fig = decomposition.plot()

plt.show()
```

A plot of time series decomposition on the CO2 data



Extracting components from time series decomposition

```
print(dir(decomposition))
```

print out the attributes associated to the decomposition variable

```
['__class__', '__delattr__', '__dict__',
 ... 'plot', 'resid', 'seasonal', 'trend']
```

```
print(decomposition.seasonal)
```

```
datestamp
1958-03-29    1.028042
1958-04-05    1.235242
1958-04-12    1.412344
1958-04-19    1.701186
```

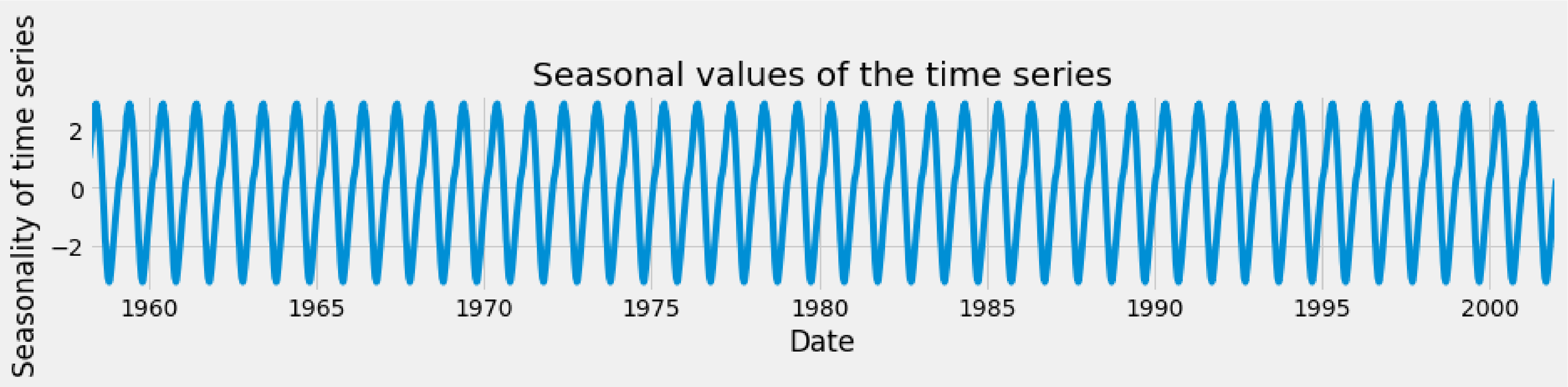
Seasonality component in time series

```
decomp_seasonal = decomposition.seasonal

ax = decomp_seasonal.plot(figsize=(14, 2))
ax.set_xlabel('Date')
ax.set_ylabel('Seasonality of time series')
ax.set_title('Seasonal values of the time series')

plt.show()
```

Seasonality component in time series



A seasonal pattern exists when a time series is influenced by seasonal factors.
Seasonality should always be a fixed and known period.

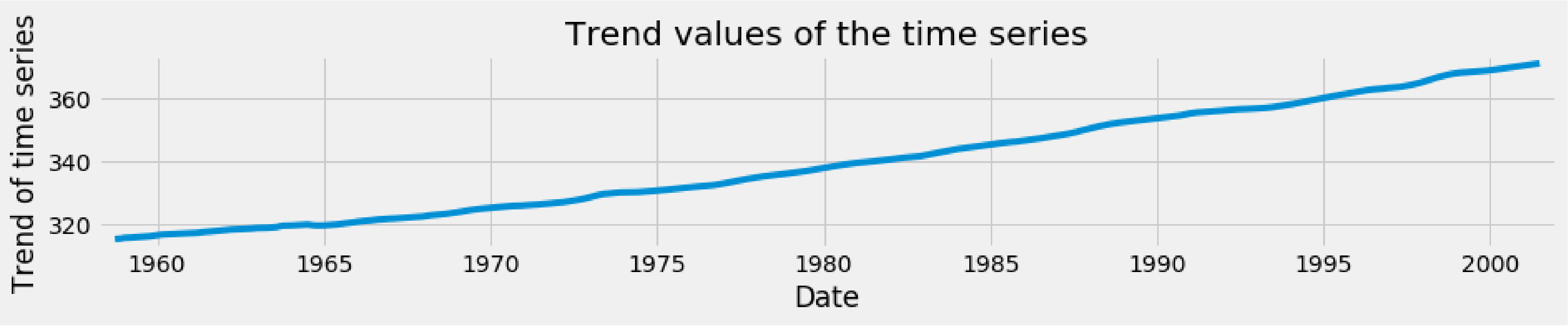
Trend component in time series

```
decomp_trend = decomposition.trend

ax = decomp_trend.plot(figsize=(14, 2))
ax.set_xlabel('Date')
ax.set_ylabel('Trend of time series')
ax.set_title('Trend values of the time series')

plt.show()
```

Trend component in time series



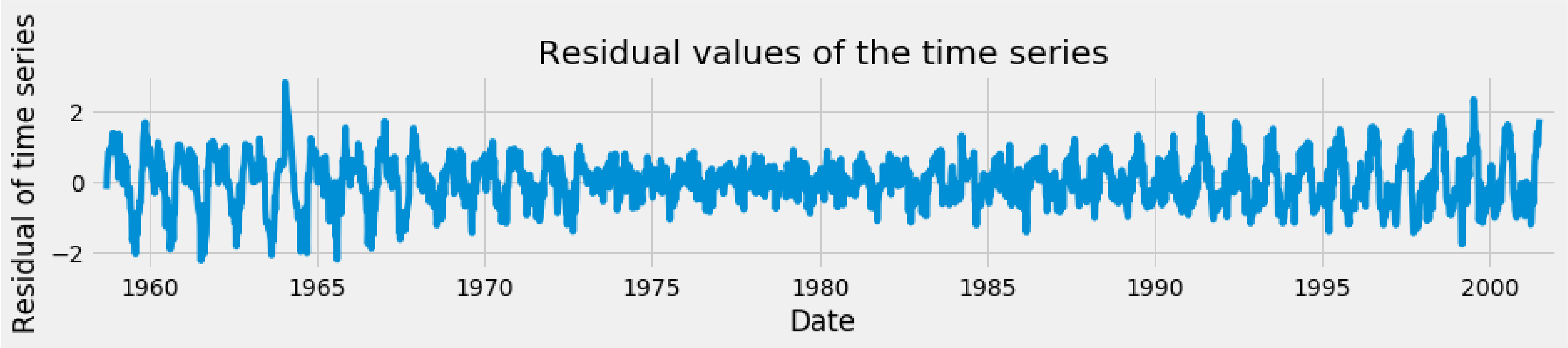
Noise component in time series

```
decomp_resid = decomp.resid

ax = decomp_resid.plot(figsize=(14, 2))
ax.set_xlabel('Date')
ax.set_ylabel('Residual of time series')
ax.set_title('Residual values of the time series')

plt.show()
```

Noise component in time series



Let's practice!

VISUALIZING TIME SERIES DATA IN PYTHON

A review on what you have learned so far

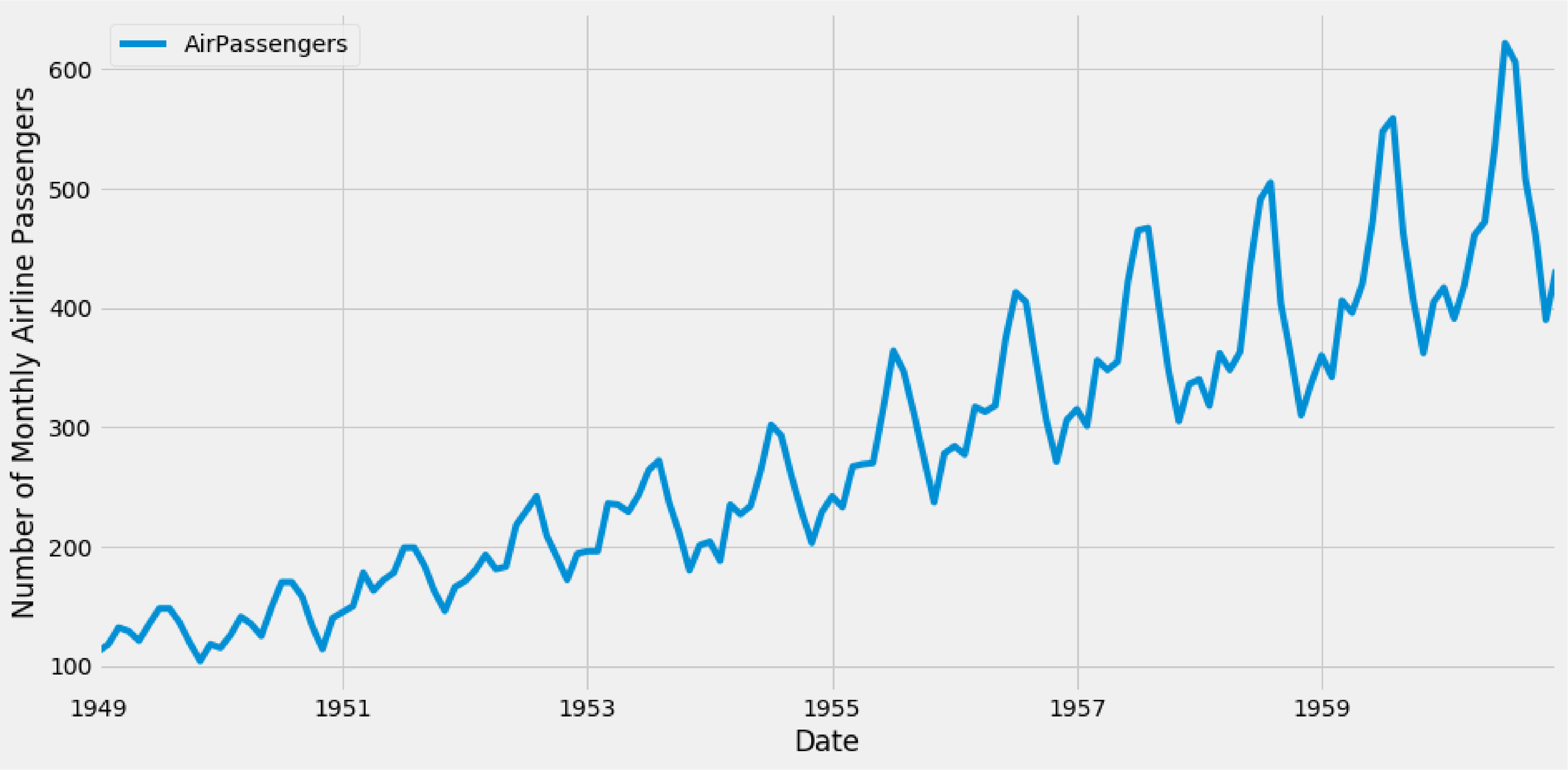
VISUALIZING TIME SERIES DATA IN PYTHON

Thomas Vincent
Head of Data Science, Getty Images

So far ...

- Visualize aggregates of time series data
- Extract statistical summaries
- Autocorrelation and Partial autocorrelation
- Time series decomposition

The airline dataset



Let's analyze this data!

VISUALIZING TIME SERIES DATA IN PYTHON