



# Peak Finding in Python

Learn how to find peaks and valleys on datasets in Python

Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. [Sign up for early access now.](https://plotly.com/studio/?utm_medium=graphing-libraries&utm_campaign=studio_early_access&utm_content=sidebar) ([https://plotly.com/studio/?utm\\_medium=graphing-libraries&utm\\_campaign=studio\\_early\\_access&utm\\_content=sidebar](https://plotly.com/studio/?utm_medium=graphing-libraries&utm_campaign=studio_early_access&utm_content=sidebar))

## Imports

The tutorial below imports [Pandas](https://pandas.pydata.org/docs/user_guide/10min.html) ([https://pandas.pydata.org/docs/user\\_guide/10min.html](https://pandas.pydata.org/docs/user_guide/10min.html)), and [SciPy](https://www.scipy.org/) (<https://www.scipy.org/>).

```
import pandas as pd
from scipy.signal import find_peaks
```

## Import Data

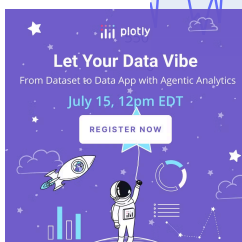
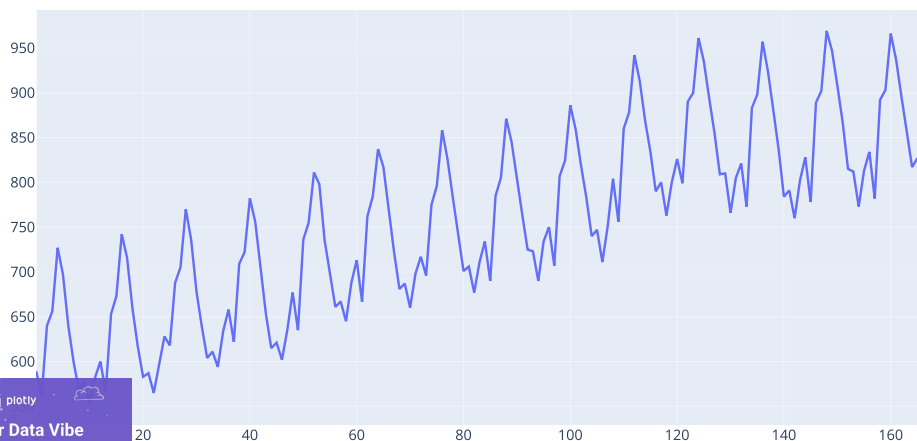
To start detecting peaks, we will import some data on milk production by month:

```
import plotly.graph_objects as go
import pandas as pd

milk_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/monthly-milk-production-pounds.csv')
time_series = milk_data['Monthly milk production (pounds per cow)']

fig = go.Figure(data=go.Scatter(
    y = time_series,
    mode = 'lines'
))

fig.show()
```



## Peak Detection

We need to find the x-axis indices for the peaks in order to determine where the peaks are located.

```
import plotly.graph_objects as go
import pandas as pd
from scipy.signal import find_peaks

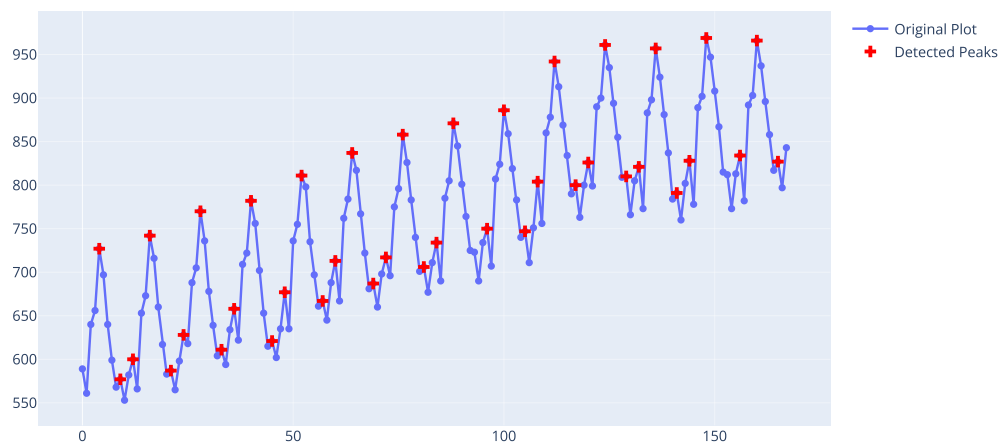
milk_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/monthly-milk-production-pounds.csv')
time_series = milk_data['Monthly milk production (pounds per cow)']

indices = find_peaks(time_series)[0]

fig = go.Figure()
fig.add_trace(go.Scatter(
    y=time_series,
    mode='lines+markers',
    name='Original Plot'
))

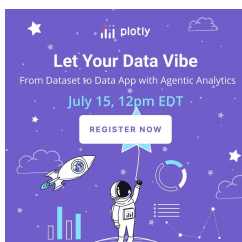
fig.add_trace(go.Scatter(
    x=indices,
    y=[time_series[j] for j in indices],
    mode='markers',
    marker=dict(
        size=8,
        color='red',
        symbol='cross'
    ),
    name='Detected Peaks'
))

fig.show()
```



## Only Highest Peaks

We can attempt to set our threshold so that we identify as many of the *highest peaks* that we can.



```

import plotly.graph_objects as go
import numpy as np
import pandas as pd
from scipy.signal import find_peaks

milk_data = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/monthly-milk-production-pounds.csv')
time_series = milk_data['Monthly milk production (pounds per cow)']

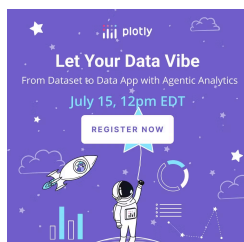
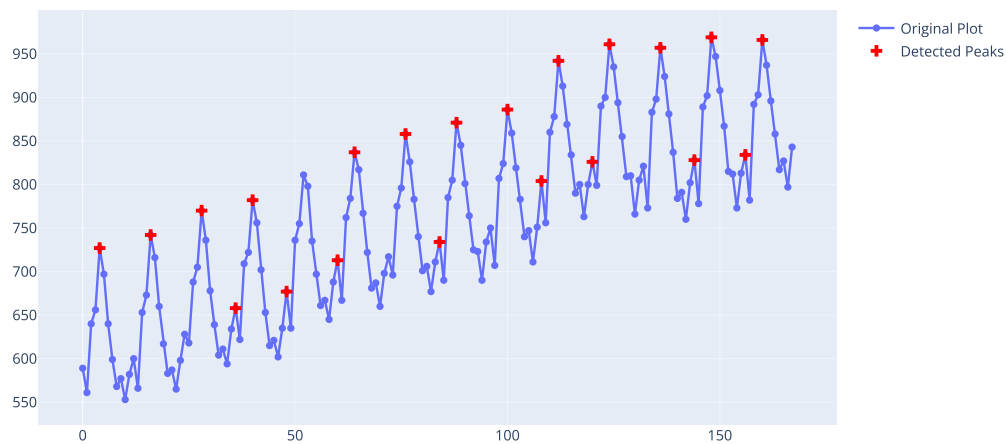
indices = find_peaks(time_series, threshold=20)[0]

fig = go.Figure()
fig.add_trace(go.Scatter(
    y=time_series,
    mode='lines+markers',
    name='Original Plot'
))

fig.add_trace(go.Scatter(
    x=indices,
    y=[time_series[j] for j in indices],
    mode='markers',
    marker=dict(
        size=8,
        color='red',
        symbol='cross'
    ),
    name='Detected Peaks'
))

fig.show()

```



# What About Dash?

Dash (<https://dash.plot.ly/>) is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at <https://dash.plot.ly/installation> (<https://dash.plot.ly/installation>).


Everywhere in this page that you see `fig.show()`, you can display the same figure in a Dash application by passing it to the `figure` argument of the `Graph` component (<https://dash.plot.ly/dash-core-components/graph>) from the built-in `dash_core_components` package like this:

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html

app = Dash()
app.layout = html.Div([
    dcc.Graph(figure=fig)
])

app.run(debug=True, use_reloader=False) # Turn off reloader if inside Jupyter
```



## Dash your way to interactive web apps.

No JavaScript required!

GET STARTED NOW


### My First App with Data, Graph, and Controls

pop

lifeExp

gdpPerCap

country	pop	continent	lifeExp	gdpPerCap
Afghanistan	31889923	Asia	43.828	974.5883384
Albania	3600523	Europe	76.423	5937.829525999999
Algeria	33333216	Africa	72.381	6223.367465
Angola	12420476	Africa	42.731	4707.231267
Argentina	40301927	Americas	75.32	12779.37964
Australia	20434176	Oceania	81.235	34435.367439999995
Austria	8199783	Europe	79.829	36126.4927
Bahrain	706573	Asia	75.635	29796.04834
Bangladesh	150448339	Asia	64.062	1761.253792
Belgium	10391226	Europe	79.441	33962.04968
Benin	8878314	Africa	56.728	1441.284873
Bolivia	9139152	Americas	65.554	3821.137884



([https://dash.plotly.com/tutorial?utm\\_medium=graphing\\_libraries&utm\\_content=python\\_footer](https://dash.plotly.com/tutorial?utm_medium=graphing_libraries&utm_content=python_footer))

### JOIN OUR MAILING LIST

Sign up to stay in the loop with all things Plotly — from Dash Club to product updates, webinars, and more!

SUBSCRIBE  
(<https://go.plot.ly/subscription>)

### About Us

Careers (<https://plotly.com/careers>)  
Resources (<https://plotly.com/resources/>)  
Blog (<https://medium.com/@plotlygraphs>)

### Products

Dash (<https://plotly.com/dash/>)  
Consulting and Training (<https://plotly.com/consulting-and-oem/>)

### Support

Community Support (<https://community.plot.ly/>)  
Documentation (<https://plotly.com/graphing-libraries>)

### Pricing

Enterprise Pricing (<https://plotly.com/get-pricing/>)

