

Log Plots in Python

How to make Log plots in Python with Plotly.

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)

This page shows examples of how to configure [2-dimensional Cartesian axes \(/python/figure-structure/#2d-cartesian-trace-types-and-subplots\)](#) to follow a logarithmic rather than linear progression. [Configuring gridlines, ticks, tick labels and axis titles \(/python/axes/\)](#) on logarithmic axes is done the same was as with [linear axes \(/python/axes/\)](#).

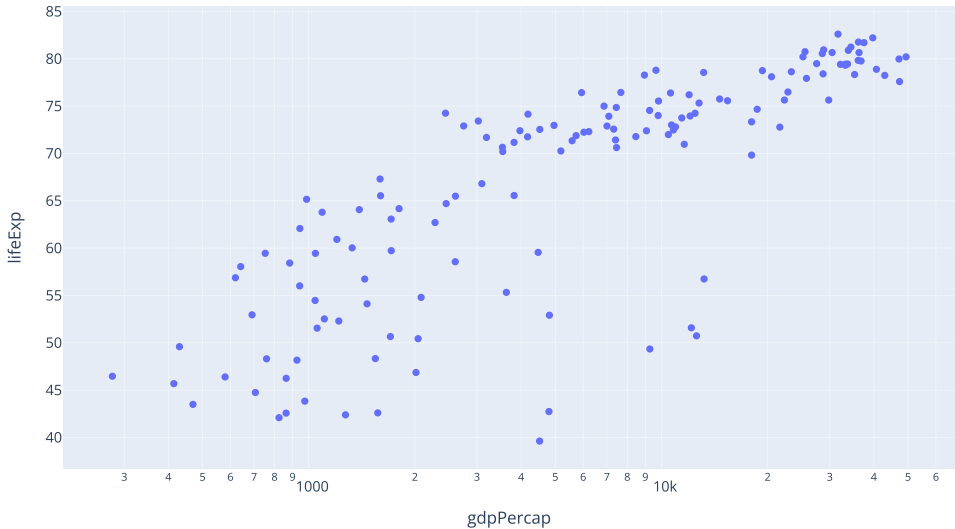
Logarithmic Axes with Plotly Express

[Plotly Express \(/python/plotly-express/\)](#) is the easy-to-use, high-level interface to Plotly, which [operates on a variety of types of data \(/python/px-arguments/\)](#) and produces [easy-to-style figures \(/python/styling-plotly-express/\)](#).

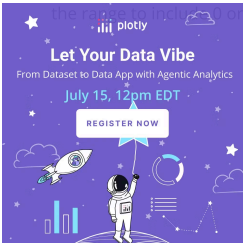
All of Plotly Express' 2-D Cartesian functions include the `log_x` and `log_y` keyword arguments, which can be set to `True` to set the corresponding axis to a logarithmic scale:

```
import plotly.express as px
df = px.data.gapminder().query("year == 2007")

fig = px.scatter(df, x="gdpPercap", y="lifeExp", hover_name="country", log_x=True)
fig.show()
```

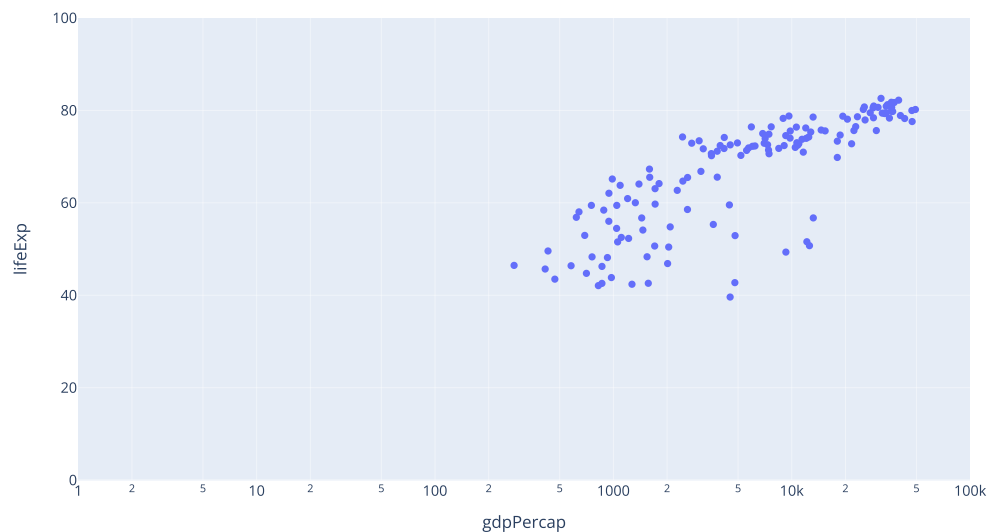


Setting the range of a logarithmic axis with Plotly Express works the same was as with linear axes: using the `range_x` and `range_y` keywords. Note that you cannot set less.



```
import plotly.express as px
df = px.data.gapminder().query("year == 2007")

fig = px.scatter(df, x="gdpPercap", y="lifeExp", hover_name="country",
                 log_x=True, range_x=[1,100000], range_y=[0,100])
fig.show()
```

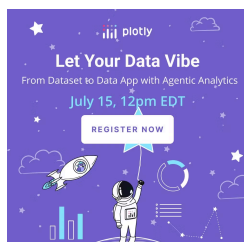


Adding minor ticks

new in 5.8

You can position and style minor ticks using `minor`. This takes a dict of properties to apply to minor ticks. See the [figure reference](https://plotly.com/python/reference/layout/xaxis/#layout-xaxis-minor) (<https://plotly.com/python/reference/layout/xaxis/#layout-xaxis-minor>) for full details on the accepted keys in this dict.

In this example we set the tick length with `ticklen`, add the ticks on the inside with `ticks="inside"`, and turn grid lines on with `showgrid=True`.

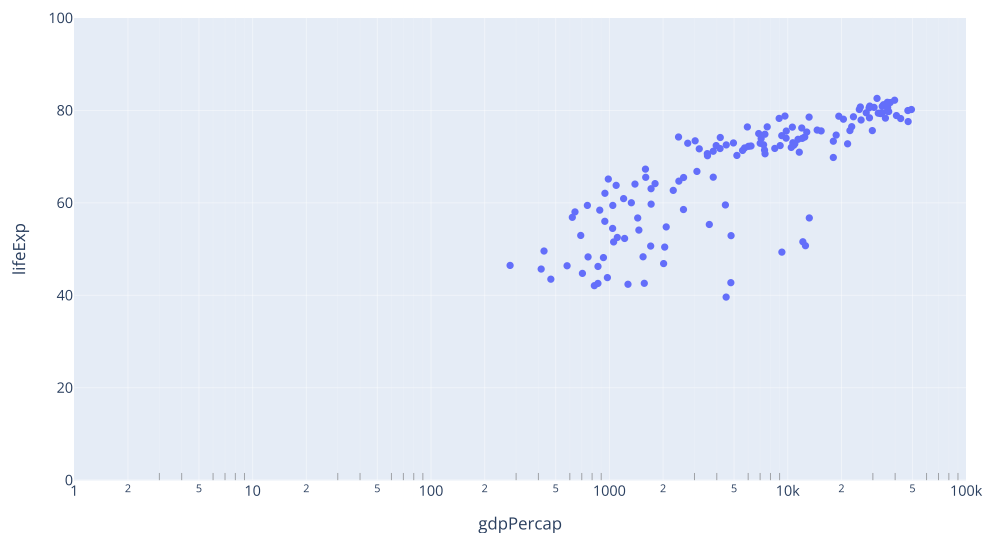


```
import plotly.express as px
df = px.data.gapminder().query("year == 2007")

fig = px.scatter(df, x="gdpPercap", y="lifeExp", hover_name="country",
                 log_x=True, range_x=[1,100000], range_y=[0,100])

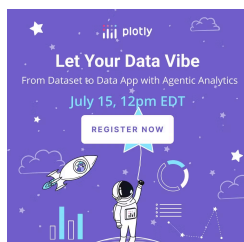
fig.update_xaxes(minor=dict(ticks="inside", ticklen=6, showgrid=True))

fig.show()
```



Logarithmic Axes with Graph Objects

If Plotly Express does not provide a good starting point, it is also possible to use [the more generic `go` Figure class from `plotly.graph_objects` \(`/python/graph-objects/`\).](https://plotly.com/python/graph-objects/)

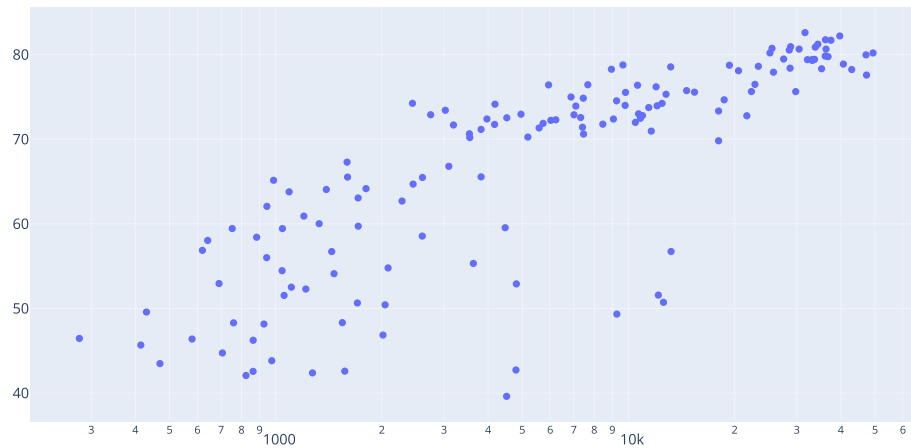


```
import plotly.graph_objects as go
import plotly.express as px
df = px.data.gapminder().query("year == 2007")

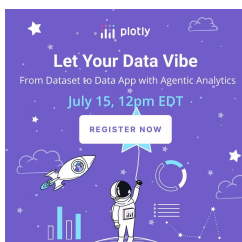
fig = go.Figure()

fig.add_trace(go.Scatter(mode="markers", x=df["gdpPercap"], y=df["lifeExp"] ))

fig.update_xaxes(type="log")
fig.show()
```



Setting the range of a logarithmic axis with `plotly.graph_objects` is *very different* than setting the range of linear axes: the range is set using the exponent rather than the actual value:



```

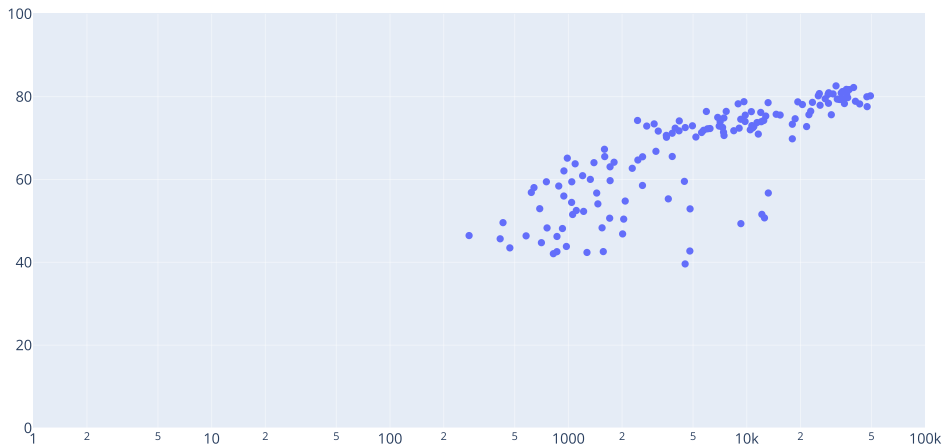
import plotly.graph_objects as go
import plotly.express as px
df = px.data.gapminder().query("year == 2007")

fig = go.Figure()

fig.add_trace(go.Scatter(mode="markers", x=df["gdpPercap"], y=df["lifeExp"] ))

fig.update_xaxes(type="log", range=[0,5]) # Log range: 10^0=1, 10^5=100000
fig.update_yaxes(range=[0,100]) # Linear range
fig.show()

```



Reference

See [function reference for px.scatter](https://plotly.com/python-api-reference/generated/plotly.express.scatter) (<https://plotly.com/python-api-reference/generated/plotly.express.scatter>) or <https://plotly.com/python/reference/layout/axis/#layout-axis-type> (<https://plotly.com/python/reference/layout/axis/#layout-axis-type>) for more information and chart attribute options!

What About Dash?

[Dash](https://dash.plot.ly/) (<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) (<https://dash.plot.ly/dash-core-components/graph>) from the built-in `dash_core_components` package like this:

```

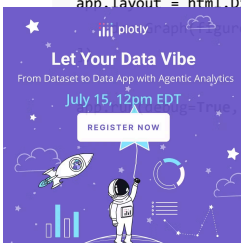
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )


from dash import Dash, dcc, html

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

    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	2600522	Europe	76.422	5937.625525999999
Algeria	33333216	Africa	72.361	6223.367465
Angola	12428676	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	708573	Asia	75.635	29796.04854
Bangladesh	158448339	Asia	64.062	1501.253792
Belgium	10592226	Europe	79.441	33692.04908
Benin	8078314	Africa	56.728	1441.284873
Bolivia	9119152	Americas	65.554	3822.137884



continent	avg lifeExp
Asia	~65
Europe	~75
Africa	~55
Americas	~70
Oceania	~78

(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/>)

Pricing

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

Support

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

