matplotlib.pyplot

`matplotlib.pyplot` is a state-based interface to matplotlib. It provides an implicit, MATLAB-like, way of plotting. It also opens figures on your screen, and acts as the figure GUI manager.

pyplot is mainly intended for interactive plots and simple cases of programmatic plot generation:

```
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 5, 0.1)
y = np.sin(x)
plt.plot(x, y)
```

The explicit (object-oriented) API is recommended for complex plots, though pyplot is still usually used to create the figure and often the axes in the figure. See `pyplot.figure`, `pyplot.subplots`, and `pyplot.subplot_mosaic` to create figures, and Axes API for the plotting methods on an axes:

```
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 5, 0.1)
y = np.sin(x)
fig, ax = plt.subplots()
ax.plot(x, y)
```

Functions

`acorr`(x, *[, data])	Plot the autocorrelation of *x*.
`angle_spectrum`(x[, Fs, Fc, window, pad_to, ...])	Plot the angle spectrum.
`annotate`(text, xy, *args, **kwargs)	Annotate the point *xy* with text *text*.
`arrow`(x, y, dx, dy, **kwargs)	Add an arrow to the Axes.
`autoscale`([enable, axis, tight])	Autoscale the axis view to the data (toggle).
`autumn`()	Set the colormap to 'autumn'.
`axes`([arg])	Add an axes to the current figure and make it the current axes.
`axhline`([y, xmin, xmax])	Add a horizontal line across the axis.
`axhspan`(ymin, ymax[, xmin, xmax])	Add a horizontal span (rectangle) across the Axes.
`axis`(*args[, emit])	Convenience method to get or set some axis properties.
`axline`(xy1[, xy2, slope])	Add an infinitely long straight line.
`axvline`([x, ymin, ymax])	Add a vertical line across the Axes.
`axvspan`(xmin, xmax[, ymin, ymax])	Add a vertical span (rectangle) across the Axes.
`bar`(x, height[, width, bottom, align, data])	Make a bar plot.

`bar_label`(container[, labels, fmt, ...])	Label a bar plot.
`barbs`(*args[, data])	Plot a 2D field of barbs.
`barh`(y, width[, height, left, align])	Make a horizontal bar plot.
`bone`()	Set the colormap to 'bone'.
`box`([on])	Turn the axes box on or off on the current axes.
`boxplot`(x[, notch, sym, vert, whis, ...])	Draw a box and whisker plot.
`broken_barh`(xranges, yrange, *[, data])	Plot a horizontal sequence of rectangles.
`cla`()	Clear the current axes.
`clabel`(CS[, levels])	Label a contour plot.
`clf`()	Clear the current figure.
`clim`([vmin, vmax])	Set the color limits of the current image.
`close`([fig])	Close a figure window.
`cohere`(x, y[, NFFT, Fs, Fc, detrend, ...])	Plot the coherence between *x* and *y*.
`colorbar`([mappable, cax, ax])	Add a colorbar to a plot.
`connect`(s, func)	Bind function *func* to event *s*.
`contour`(*args[, data])	Plot contour lines.
`contourf`(*args[, data])	Plot filled contours.
`cool`()	Set the colormap to 'cool'.
`copper`()	Set the colormap to 'copper'.
`csd`(x, y[, NFFT, Fs, Fc, detrend, window, ...])	Plot the cross-spectral density.
`delaxes`([ax])	Remove an `Axes` (defaulting to the current axes) from its figure.
`disconnect`(cid)	Disconnect the callback with id *cid*.
`draw`()	Redraw the current figure.
`draw_if_interactive`()	Redraw the current figure if in interactive mode.
`errorbar`(x, y[, yerr, xerr, fmt, ecolor, ...])	Plot y versus x as lines and/or markers with attached errorbars.
`eventplot`(positions[, orientation, ...])	Plot identical parallel lines at the given positions.
`figimage`(X[, xo, yo, alpha, norm, cmap, ...])	Add a non-resampled image to the figure.
`figlegend`(*args, **kwargs)	Place a legend on the figure.
`fignum_exists`(num)	Return whether the figure with the given id exists.
`figtext`(x, y, s[, fontdict])	Add text to figure.

figure([num, figsize, dpi, facecolor, ...])	Create a new figure, or activate an existing figure.
fill(*args[, data])	Plot filled polygons.
fill_between(x, y1[, y2, where, ...])	Fill the area between two horizontal curves.
fill_betweenx(y, x1[, x2, where, step, ...])	Fill the area between two vertical curves.
findobj([o, match, include_self])	Find artist objects.
flag()	Set the colormap to 'flag'.
gca(**kwargs)	Get the current Axes.
gcf()	Get the current figure.
gci()	Get the current colorable artist.
get(obj, *args, **kwargs)	Return the value of an **Artist**'s *property*, or print all of them.
get_current_fig_manager()	Return the figure manager of the current figure.
get_figlabels()	Return a list of existing figure labels.
get_fignums()	Return a list of existing figure numbers.
get_plot_commands()	Get a sorted list of all of the plotting commands.
getp(obj, *args, **kwargs)	Return the value of an **Artist**'s *property*, or print all of them.
ginput([n, timeout, show_clicks, mouse_add, ...])	Blocking call to interact with a figure.
gray()	Set the colormap to 'gray'.
grid([visible, which, axis])	Configure the grid lines.
hexbin(x, y[, C, gridsize, bins, xscale, ...])	Make a 2D hexagonal binning plot of points *x*, *y*.
hist(x[, bins, range, density, weights, ...])	Plot a histogram.
hist2d(x, y[, bins, range, density, ...])	Make a 2D histogram plot.
hlines(y, xmin, xmax[, colors, linestyles, ...])	Plot horizontal lines at each *y* from *xmin* to *xmax*.
hot()	Set the colormap to 'hot'.
hsv()	Set the colormap to 'hsv'.
imread(fname[, format])	Read an image from a file into an array.
imsave(fname, arr, **kwargs)	Save an array as an image file.
imshow(X[, cmap, norm, aspect, ...])	Display data as an image, i.e., on a 2D regular raster.
inferno()	Set the colormap to 'inferno'.
install_repl_displayhook()	Install a repl display hook so that any stale figure are automatically redrawn when control is returned to the repl.

Search the docs ...

matplotlib.path
matplotlib.patheffects
matplotlib.pyplot
matplotlib.pyplot
matplotlib.pyplot.acorr
matplotlib.pyplot.angle_spec
matplotlib.pyplot.annotate
matplotlib.pyplot.arrow
matplotlib.pyplot.autoscale
matplotlib.pyplot.autumn
matplotlib.pyplot.axes
matplotlib.pyplot.axhline
matplotlib.pyplot.axhspan
matplotlib.pyplot.axis
matplotlib.pyplot.axline
matplotlib.pyplot.axvline
matplotlib.pyplot.axvspan
matplotlib.pyplot.bar
matplotlib.pyplot.bar_label
matplotlib.pyplot.barbs
matplotlib.pyplot.barh
matplotlib.pyplot.bone
matplotlib.pyplot.box
matplotlib.pyplot.boxplot
matplotlib.pyplot.broken_bar
matplotlib.pyplot.cla
matplotlib.pyplot.clabel
matplotlib.pyplot.clf
matplotlib.pyplot.clim
matplotlib.pyplot.close
matplotlib.pyplot.cohere
matplotlib.pyplot.colorbar
matplotlib.pyplot.connect
matplotlib.pyplot.contour
matplotlib.pyplot.contourf
matplotlib.pyplot.cool
matplotlib.pyplot.copper
matplotlib.pyplot.csd
matplotlib.pyplot.delaxes
matplotlib.pyplot.disconnect
matplotlib.pyplot.draw
matplotlib.pyplot.draw_if_inte

matplotlib.pyplot.errorbar
matplotlib.pyplot.eventplot
matplotlib.pyplot.figimage

ioff()	Disable interactive mode.
ion()	Enable interactive mode.
isinteractive()	Return whether plots are updated after every plotting command.
jet()	Set the colormap to 'jet'.
legend(*args, **kwargs)	Place a legend on the Axes.
locator_params([axis, tight])	Control behavior of major tick locators.
loglog(*args, **kwargs)	Make a plot with log scaling on both the x and y axis.
magma()	Set the colormap to 'magma'.
magnitude_spectrum(x[, Fs, Fc, window, ...])	Plot the magnitude spectrum.
margins(*margins[, x, y, tight])	Set or retrieve autoscaling margins.
matshow(A[, fignum])	Display an array as a matrix in a new figure window.
minorticks_off()	Remove minor ticks from the Axes.
minorticks_on()	Display minor ticks on the Axes.
new_figure_manager(num, *args, **kwargs)	Create a new figure manager instance.
nipy_spectral()	Set the colormap to 'nipy_spectral'.
pause(interval)	Run the GUI event loop for *interval* seconds.
pcolor(*args[, shading, alpha, norm, cmap, ...])	Create a pseudocolor plot with a non-regular rectangular grid.
pcolormesh(*args[, alpha, norm, cmap, vmin, ...])	Create a pseudocolor plot with a non-regular rectangular grid.
phase_spectrum(x[, Fs, Fc, window, pad_to, ...])	Plot the phase spectrum.
pie(x[, explode, labels, colors, autopct, ...])	Plot a pie chart.
pink()	Set the colormap to 'pink'.
plasma()	Set the colormap to 'plasma'.
plot(*args[, scalex, scaley, data])	Plot y versus x as lines and/or markers.
plot_date(x, y[, fmt, tz, xdate, ydate, data])	Plot coercing the axis to treat floats as dates.
polar(*args, **kwargs)	Make a polar plot.
prism()	Set the colormap to 'prism'.
psd(x[, NFFT, Fs, Fc, detrend, window, ...])	Plot the power spectral density.
quiver(*args[, data])	Plot a 2D field of arrows.
quiverkey(Q, X, Y, U, label, **kwargs)	Add a key to a quiver plot.

`rc`(group, **kwargs)	Set the current `rcParams`. *group* is the grouping for the rc, e.g., for `lines.linewidth` the group is `lines`, for `axes.facecolor`, the group is `axes`, and so on. Group may also be a list or tuple of group names, e.g., (*xtick*, *ytick*). *kwargs* is a dictionary attribute name/value pairs, e.g.,::.
`rc_context`([rc, fname])	Return a context manager for temporarily changing rcParams.
`rcdefaults`()	Restore the `rcParams` from Matplotlib's internal default style.
`rgrids`([radii, labels, angle, fmt])	Get or set the radial gridlines on the current polar plot.
`savefig`(*args, **kwargs)	Save the current figure.
`sca`(ax)	Set the current Axes to *ax* and the current Figure to the parent of *ax*.
`scatter`(x, y[, s, c, marker, cmap, norm, ...])	A scatter plot of *y* vs.
`sci`(im)	Set the current image.
`semilogx`(*args, **kwargs)	Make a plot with log scaling on the x axis.
`semilogy`(*args, **kwargs)	Make a plot with log scaling on the y axis.
`set_cmap`(cmap)	Set the default colormap, and applies it to the current image if any.
`set_loglevel`(*args, **kwargs)	Set Matplotlib's root logger and root logger handler level, creating the handler if it does not exist yet.
`setp`(obj, *args, **kwargs)	Set one or more properties on an `Artist`, or list allowed values.
`show`(*[, block])	Display all open figures.
`specgram`(x[, NFFT, Fs, Fc, detrend, window, ...])	Plot a spectrogram.
`spring`()	Set the colormap to 'spring'.
`spy`(Z[, precision, marker, markersize, ...])	Plot the sparsity pattern of a 2D array.
`stackplot`(x, *args[, labels, colors, ...])	Draw a stacked area plot.
`stairs`(values[, edges, orientation, ...])	A stepwise constant function as a line with bounding edges or a filled plot.
`stem`(*args[, linefmt, markerfmt, basefmt, ...])	Create a stem plot.
`step`(x, y, *args[, where, data])	Make a step plot.
`streamplot`(x, y, u, v[, density, linewidth, ...])	Draw streamlines of a vector flow.
`subplot`(*args, **kwargs)	Add an Axes to the current figure or retrieve an existing Axes.

`subplot2grid`(shape, loc[, rowspan, colspan, fig])	Create a subplot at a specific location inside a regular grid.
`subplot_mosaic`(mosaic, *[, sharex, sharey, ...])	Build a layout of Axes based on ASCII art or nested lists.
`subplot_tool`([targetfig])	Launch a subplot tool window for a figure.
`subplots`([nrows, ncols, sharex, sharey, ...])	Create a figure and a set of subplots.
`subplots_adjust`([left, bottom, right, top, ...])	Adjust the subplot layout parameters.
`summer`()	Set the colormap to 'summer'.
`suptitle`(t, **kwargs)	Add a centered suptitle to the figure.
`switch_backend`(newbackend)	Close all open figures and set the Matplotlib backend.
`table`([cellText, cellColours, cellLoc, ...])	Add a table to an `Axes`.
`text`(x, y, s[, fontdict])	Add text to the Axes.
`thetagrids`([angles, labels, fmt])	Get or set the theta gridlines on the current polar plot.
`tick_params`([axis])	Change the appearance of ticks, tick labels, and gridlines.
`ticklabel_format`(*[, axis, style, ...])	Configure the `ScalarFormatter` used by default for linear axes.
`tight_layout`(*[, pad, h_pad, w_pad, rect])	Adjust the padding between and around subplots.
`title`(label[, fontdict, loc, pad, y])	Set a title for the Axes.
`tricontour`(*args, **kwargs)	Draw contour lines on an unstructured triangular grid.
`tricontourf`(*args, **kwargs)	Draw contour regions on an unstructured triangular grid.
`tripcolor`(*args[, alpha, norm, cmap, vmin, ...])	Create a pseudocolor plot of an unstructured triangular grid.
`triplot`(*args, **kwargs)	Draw a unstructured triangular grid as lines and/or markers.
`twinx`([ax])	Make and return a second axes that shares the *x*-axis.
`twiny`([ax])	Make and return a second axes that shares the *y*-axis.
`uninstall_repl_displayhook`()	Uninstall the Matplotlib display hook.
`violinplot`(dataset[, positions, vert, ...])	Make a violin plot.
`viridis`()	Set the colormap to 'viridis'.
`vlines`(x, ymin, ymax[, colors, linestyles, ...])	Plot vertical lines at each *x* from *ymin* to *ymax*.
`waitforbuttonpress`([timeout])	Blocking call to interact with the figure.
`winter`()	Set the colormap to 'winter'.

`xcorr`(x, y[, normed, detrend, usevlines, ...])	Plot the cross correlation between *x* and *y*.
`xkcd`([scale, length, randomness])	Turn on xkcd sketch-style drawing mode.
`xlabel`(xlabel[, fontdict, labelpad, loc])	Set the label for the x-axis.
`xlim`(*args, **kwargs)	Get or set the x limits of the current axes.
`xscale`(value, **kwargs)	Set the x-axis scale.
`xticks`([ticks, labels])	Get or set the current tick locations and labels of the x-axis.
`ylabel`(ylabel[, fontdict, labelpad, loc])	Set the label for the y-axis.
`ylim`(*args, **kwargs)	Get or set the y-limits of the current axes.
`yscale`(value, **kwargs)	Set the y-axis scale.
`yticks`([ticks, labels])	Get or set the current tick locations and labels of the y-axis.

Created using Sphinx 4.3.0.