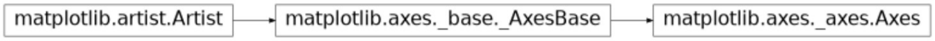
Search the docs ...

matplotlib.axes

Table of Contents

Inheritance



The Axes class

class matplotlib.axes.**Axes**(*fig*, *rect*, ***, *facecolor=None*, *frameon=True*, *sharex=None*, *sharey=None*, *label=''*, *xscale=None*, *yscale=None*, *box_aspect=None*, ***kwargs*) [source]

Bases: `matplotlib.axes._base._AxesBase`

The `Axes` contains most of the figure elements: `Axis`, `Tick`, `Line2D`, `Text`, `Polygon`, etc., and sets the coordinate system.

The `Axes` instance supports callbacks through a callbacks attribute which is a `CallbackRegistry` instance. The events you can connect to are 'xlim_changed' and 'ylim_changed' and the callback will be called with func(*ax*) where *ax* is the `Axes` instance.

Attributes: **dataLim** : *Bbox*

The bounding box enclosing all data displayed in the Axes.

viewLim : *Bbox*

The view limits in data coordinates.

Build an Axes in a figure.

Parameters: **fig** : *Figure*

The Axes is built in the `Figure` *fig*.

rect : ***[left, bottom, width, height]***

The Axes is built in the rectangle *rect*. *rect* is in `Figure` coordinates.

sharex, sharey : ***Axes*, *optional***

The x or y `axis` is shared with the x or y axis in the input `Axes`.

frameon : ***bool, default: True***

Whether the Axes frame is visible.

box_aspect : ***float, optional***

Set a fixed aspect for the Axes box, i.e. the ratio of height to width. See `set_box_aspect` for details.

****kwargs**

Other optional keyword arguments:

Property	Description
`adjustable`	{'box', 'datalim'}
`agg_filter`	a filter function, which takes a (m, n, 3) float array and a dpi value, and returns a (m, n, 3) array
`alpha`	scalar or None
`anchor`	(float, float) or {'C', 'SW', 'S', 'SE', 'E', 'NE', ...}
`animated`	bool
`aspect`	{'auto', 'equal'} or float
`autoscale_on`	bool
`autoscalex_on`	bool
`autoscaley_on`	bool
`axes_locator`	Callable[[Axes, Renderer], Bbox]
`axisbelow`	bool or 'line'
`box_aspect`	float or None
`clip_box`	`Bbox`
`clip_on`	bool
`clip_path`	Patch or (Path, Transform) or None
`facecolor` or fc	color
`figure`	`Figure`
`frame_on`	bool
`gid`	str
`in_layout`	bool
`label`	object
`navigate`	bool
`navigate_mode`	unknown
`path_effects`	`AbstractPathEffect`
`picker`	None or bool or float or callable
`position`	[left, bottom, width, height] or `Bbox`
`prop_cycle`	unknown
`rasterization_zorder`	float or None
`rasterized`	bool
`sketch_params`	(scale: float, length: float, randomness: float)

snap	bool or None
title	str
transform	Transform
url	str
visible	bool
xbound	unknown
xlabel	str
xlim	(bottom: float, top: float)
xmargin	float greater than -0.5
xscale	{"linear", "log", "symlog", "logit", ...} or ScaleBase
xticklabels	unknown
xticks	unknown
ybound	unknown
	ScaleBase
yticklabels	unknown
yticks	unknown
zorder	float

Returns: Axes

The new Axes object.

Subplots

SubplotBase	Base class for subplots, which are Axes instances with additional methods to facilitate generating and manipulating a set of Axes within a figure.
subplot_class_factory	

Plotting

Basic

Axes.plot	Plot y versus x as lines and/or markers.
Axes.errorbar	Plot y versus x as lines and/or markers with attached errorbars.
Axes.scatter	A scatter plot of *y* vs.
Axes.plot_date	Plot coercing the axis to treat floats as dates.
Axes.step	Make a step plot.
Axes.loglog	Make a plot with log scaling on both the x and y axis.
Axes.semilogx	Make a plot with log scaling on the x axis.

`Axes.semilogy`	Make a plot with log scaling on the y axis.
`Axes.fill_between`	Fill the area between two horizontal curves.
`Axes.fill_betweenx`	Fill the area between two vertical curves.
`Axes.bar`	Make a bar plot.
`Axes.barh`	Make a horizontal bar plot.
`Axes.bar_label`	Label a bar plot.
`Axes.stem`	Create a stem plot.
`Axes.eventplot`	Plot identical parallel lines at the given positions.
`Axes.pie`	Plot a pie chart.
`Axes.stackplot`	Draw a stacked area plot.
`Axes.broken_barh`	Plot a horizontal sequence of rectangles.
`Axes.vlines`	Plot vertical lines at each *x* from *ymin* to *ymax*.
`Axes.hlines`	Plot horizontal lines at each *y* from *xmin* to *xmax*.
`Axes.fill`	Plot filled polygons.

Spans

`Axes.axhline`	Add a horizontal line across the axis.
`Axes.axhspan`	Add a horizontal span (rectangle) across the Axes.
`Axes.axvline`	Add a vertical line across the Axes.
`Axes.axvspan`	Add a vertical span (rectangle) across the Axes.
`Axes.axline`	Add an infinitely long straight line.

Spectral

`Axes.acorr`	Plot the autocorrelation of *x*.
`Axes.angle_spectrum`	Plot the angle spectrum.
`Axes.cohere`	Plot the coherence between *x* and *y*.
`Axes.csd`	Plot the cross-spectral density.
`Axes.magnitude_spectrum`	Plot the magnitude spectrum.
`Axes.phase_spectrum`	Plot the phase spectrum.
`Axes.psd`	Plot the power spectral density.
`Axes.specgram`	Plot a spectrogram.

`Axes.xcorr`	Plot the cross correlation between *x* and *y*.

Statistics

`Axes.boxplot`	Draw a box and whisker plot.
`Axes.violinplot`	Make a violin plot.
`Axes.violin`	Drawing function for violin plots.
`Axes.bxp`	Drawing function for box and whisker plots.

Binned

`Axes.hexbin`	Make a 2D hexagonal binning plot of points *x*, *y*.
`Axes.hist`	Plot a histogram.
`Axes.hist2d`	Make a 2D histogram plot.
`Axes.stairs`	A stepwise constant function as a line with bounding edges or a filled plot.

Contours

`Axes.clabel`	Label a contour plot.
`Axes.contour`	Plot contour lines.
`Axes.contourf`	Plot filled contours.

2D arrays

`Axes.imshow`	Display data as an image, i.e., on a 2D regular raster.
`Axes.matshow`	Plot the values of a 2D matrix or array as color-coded image.
`Axes.pcolor`	Create a pseudocolor plot with a non-regular rectangular grid.
`Axes.pcolorfast`	Create a pseudocolor plot with a non-regular rectangular grid.
`Axes.pcolormesh`	Create a pseudocolor plot with a non-regular rectangular grid.
`Axes.spy`	Plot the sparsity pattern of a 2D array.

Unstructured triangles

`Axes.tripcolor`	Create a pseudocolor plot of an unstructured triangular grid.
`Axes.triplot`	Draw a unstructured triangular grid as lines and/or markers.
`Axes.tricontour`	Draw contour lines on an unstructured triangular grid.
`Axes.tricontourf`	Draw contour regions on an unstructured triangular grid.

Text and annotations

`Axes.annotate`	Annotate the point *xy* with text *text*.
`Axes.text`	Add text to the Axes.
`Axes.table`	Add a table to an `Axes`.
`Axes.arrow`	Add an arrow to the Axes.
`Axes.inset_axes`	Add a child inset Axes to this existing Axes.
`Axes.indicate_inset`	Add an inset indicator to the Axes.
`Axes.indicate_inset_zoom`	Add an inset indicator rectangle to the Axes based on the axis limits for an *inset_ax* and draw connectors between *inset_ax* and the rectangle.
`Axes.secondary_xaxis`	Add a second x-axis to this Axes.
`Axes.secondary_yaxis`	Add a second y-axis to this Axes.

Vector fields

`Axes.barbs`	Plot a 2D field of barbs.
`Axes.quiver`	Plot a 2D field of arrows.
`Axes.quiverkey`	Add a key to a quiver plot.
`Axes.streamplot`	Draw streamlines of a vector flow.

Clearing

`Axes.cla`	Clear the Axes.
`Axes.clear`	Clear the Axes.

Appearance

`Axes.axis`	Convenience method to get or set some axis properties.
`Axes.set_axis_off`	Turn the x- and y-axis off.
`Axes.set_axis_on`	Turn the x- and y-axis on.
`Axes.set_frame_on`	Set whether the Axes rectangle patch is drawn.
`Axes.get_frame_on`	Get whether the Axes rectangle patch is drawn.
`Axes.set_axisbelow`	Set whether axis ticks and gridlines are above or below most artists.
`Axes.get_axisbelow`	Get whether axis ticks and gridlines are above or below most artists.
`Axes.grid`	Configure the grid lines.

`Axes.get_facecolor`	Get the facecolor of the Axes.
`Axes.set_facecolor`	Set the facecolor of the Axes.

Property cycle

`Axes.set_prop_cycle`	Set the property cycle of the Axes.

Axis / limits

`Axes.get_xaxis`	Return the XAxis instance.
`Axes.get_yaxis`	Return the YAxis instance.

Axis limits and direction

`Axes.invert_xaxis`	Invert the x-axis.
`Axes.xaxis_inverted`	Return whether the xaxis is oriented in the "inverse" direction.
`Axes.invert_yaxis`	Invert the y-axis.
`Axes.yaxis_inverted`	Return whether the yaxis is oriented in the "inverse" direction.
`Axes.set_xlim`	Set the x-axis view limits.
`Axes.get_xlim`	Return the x-axis view limits.
`Axes.set_ylim`	Set the y-axis view limits.
`Axes.get_ylim`	Return the y-axis view limits.
`Axes.update_datalim`	Extend the `dataLim` Bbox to include the given points.
`Axes.set_xbound`	Set the lower and upper numerical bounds of the x-axis.
`Axes.get_xbound`	Return the lower and upper x-axis bounds, in increasing order.
`Axes.set_ybound`	Set the lower and upper numerical bounds of the y-axis.
`Axes.get_ybound`	Return the lower and upper y-axis bounds, in increasing order.

Axis labels, title, and legend

`Axes.set_xlabel`	Set the label for the x-axis.
`Axes.get_xlabel`	Get the xlabel text string.
`Axes.set_ylabel`	Set the label for the y-axis.
`Axes.get_ylabel`	Get the ylabel text string.
`Axes.set_title`	Set a title for the Axes.

Axes.get_title	Get an Axes title.
Axes.legend	Place a legend on the Axes.
Axes.get_legend	Return the Legend instance, or None if no legend is defined.
Axes.get_legend_handles_labels	Return handles and labels for legend

Axis scales

Axes.set_xscale	Set the x-axis scale.
Axes.get_xscale	Return the xaxis' scale (as a str).
Axes.set_yscale	Set the y-axis scale.
Axes.get_yscale	Return the yaxis' scale (as a str).

Autoscaling and margins

Axes.use_sticky_edges	When autoscaling, whether to obey all Artist.sticky_edges.
Axes.margins	Set or retrieve autoscaling margins.
Axes.set_xmargin	Set padding of X data limits prior to autoscaling.
Axes.set_ymargin	Set padding of Y data limits prior to autoscaling.
Axes.relim	Recompute the data limits based on current artists.
Axes.autoscale	Autoscale the axis view to the data (toggle).
Axes.autoscale_view	Autoscale the view limits using the data limits.
Axes.set_autoscale_on	Set whether autoscaling is applied to each axis on the next draw or call to Axes.autoscale_view.
Axes.get_autoscale_on	Return True if each axis is autoscaled, False otherwise.
Axes.set_autoscalex_on	Set whether the x-axis is autoscaled on the next draw or call to Axes.autoscale_view.
Axes.get_autoscalex_on	Return whether the x-axis is autoscaled.
Axes.set_autoscaley_on	Set whether the y-axis is autoscaled on the next draw or call to Axes.autoscale_view.
Axes.get_autoscaley_on	Return whether the y-axis is autoscaled.

Aspect ratio

Axes.apply_aspect	Adjust the Axes for a specified data aspect ratio.
Axes.set_aspect	Set the aspect ratio of the axes scaling, i.e. y/x-scale.
Axes.get_aspect	Return the aspect ratio of the axes scaling.

`Axes.set_box_aspect`	Set the Axes box aspect, i.e. the ratio of height to width.
`Axes.get_box_aspect`	Return the Axes box aspect, i.e. the ratio of height to width.
`Axes.set_adjustable`	Set how the Axes adjusts to achieve the required aspect ratio.
`Axes.get_adjustable`	Return whether the Axes will adjust its physical dimension ('box') or its data limits ('datalim') to achieve the desired aspect ratio.

Ticks and tick labels

`Axes.set_xticks`	Set the xaxis' tick locations and optionally labels.
`Axes.get_xticks`	Return the xaxis' tick locations in data coordinates.
`Axes.set_xticklabels`	Set the xaxis' labels with list of string labels.
`Axes.get_xticklabels`	Get the xaxis' tick labels.
`Axes.get_xmajorticklabels`	Return the xaxis' major tick labels, as a list of `Text`.
`Axes.get_xminorticklabels`	Return the xaxis' minor tick labels, as a list of `Text`.
`Axes.get_xgridlines`	Return the xaxis' grid lines as a list of `Line2D`s.
`Axes.get_xticklines`	Return the xaxis' tick lines as a list of `Line2D`s.
`Axes.xaxis_date`	Set up axis ticks and labels to treat data along the xaxis as dates.
`Axes.set_yticks`	Set the yaxis' tick locations and optionally labels.
`Axes.get_yticks`	Return the yaxis' tick locations in data coordinates.
`Axes.set_yticklabels`	Set the yaxis' labels with list of string labels.
`Axes.get_yticklabels`	Get the yaxis' tick labels.
`Axes.get_ymajorticklabels`	Return the yaxis' major tick labels, as a list of `Text`.
`Axes.get_yminorticklabels`	Return the yaxis' minor tick labels, as a list of `Text`.
`Axes.get_ygridlines`	Return the yaxis' grid lines as a list of `Line2D`s.
`Axes.get_yticklines`	Return the yaxis' tick lines as a list of `Line2D`s.
`Axes.yaxis_date`	Set up axis ticks and labels to treat data along the yaxis as dates.
`Axes.minorticks_off`	Remove minor ticks from the Axes.
`Axes.minorticks_on`	Display minor ticks on the Axes.
`Axes.ticklabel_format`	Configure the `ScalarFormatter` used by default for linear axes.
`Axes.tick_params`	Change the appearance of ticks, tick labels, and gridlines.
`Axes.locator_params`	Control behavior of major tick locators.

Units

`Axes.convert_xunits`	Convert *x* using the unit type of the xaxis.
`Axes.convert_yunits`	Convert *y* using the unit type of the yaxis.
`Axes.have_units`	Return whether units are set on any axis.

Adding artists

`Axes.add_artist`	Add an `Artist` to the Axes; return the artist.
`Axes.add_child_axes`	Add an `AxesBase` to the Axes' children; return the child Axes.
`Axes.add_collection`	Add a `Collection` to the Axes; return the collection.
`Axes.add_container`	Add a `Container` to the axes' containers; return the container.
`Axes.add_image`	Add an `AxesImage` to the Axes; return the image.
`Axes.add_line`	Add a `Line2D` to the Axes; return the line.
`Axes.add_patch`	Add a `Patch` to the Axes; return the patch.
`Axes.add_table`	Add a `Table` to the Axes; return the table.

Twinning and sharing

`Axes.twinx`	Create a twin Axes sharing the xaxis.
`Axes.twiny`	Create a twin Axes sharing the yaxis.
`Axes.sharex`	Share the x-axis with *other*.
`Axes.sharey`	Share the y-axis with *other*.
`Axes.get_shared_x_axes`	Return a reference to the shared axes Grouper object for x axes.
`Axes.get_shared_y_axes`	Return a reference to the shared axes Grouper object for y axes.

Axes position

`Axes.get_anchor`	Get the anchor location.
`Axes.set_anchor`	Define the anchor location.
`Axes.get_axes_locator`	Return the axes_locator.
`Axes.set_axes_locator`	Set the Axes locator.
`Axes.reset_position`	Reset the active position to the original position.
`Axes.get_position`	Return the position of the Axes within the figure as a `Bbox`.
`Axes.set_position`	Set the Axes position.

Async/event based

`Axes.stale`	Whether the artist is 'stale' and needs to be re-drawn for the output to match the internal state of the artist.
`Axes.pchanged`	Call all of the registered callbacks.
`Axes.add_callback`	Add a callback function that will be called whenever one of the `Artist`'s properties changes.
`Axes.remove_callback`	Remove a callback based on its observer id.

Interactive

`Axes.can_pan`	Return whether this Axes supports any pan/zoom button functionality.
`Axes.can_zoom`	Return whether this Axes supports the zoom box button functionality.
`Axes.get_navigate`	Get whether the Axes responds to navigation commands.
`Axes.set_navigate`	Set whether the Axes responds to navigation toolbar commands.
`Axes.get_navigate_mode`	Get the navigation toolbar button status: 'PAN', 'ZOOM', or None.
`Axes.set_navigate_mode`	Set the navigation toolbar button status.
`Axes.start_pan`	Called when a pan operation has started.
`Axes.drag_pan`	Called when the mouse moves during a pan operation.
`Axes.end_pan`	Called when a pan operation completes (when the mouse button is up.)
`Axes.format_coord`	Return a format string formatting the *x*, *y* coordinates.
`Axes.format_cursor_data`	Return a string representation of *data*.
`Axes.format_xdata`	Return *x* formatted as an x-value.
`Axes.format_ydata`	Return *y* formatted as an y-value.
`Axes.mouseover`	If this property is set to *True*, the artist will be queried for custom context information when the mouse cursor moves over it.
`Axes.in_axes`	Return whether the given event (in display coords) is in the Axes.
`Axes.contains`	Test whether the artist contains the mouse event.
`Axes.contains_point`	Return whether *point* (pair of pixel coordinates) is inside the axes patch.
`Axes.get_cursor_data`	Return the cursor data for a given event.

Children

`Axes.get_children`	Return a list of the child `Artist`s of this `Artist`.
`Axes.get_images`	Return a list of `AxesImage`s contained by the Axes.

`Axes.get_lines`	Return a list of lines contained by the Axes.
`Axes.findobj`	Find artist objects.

Drawing

`Axes.draw`	Draw the Artist (and its children) using the given renderer.
`Axes.draw_artist`	Efficiently redraw a single artist.
`Axes.redraw_in_frame`	Efficiently redraw Axes data, but not axis ticks, labels, etc.
`Axes.get_renderer_cache`	
`Axes.get_rasterization_zorder`	Return the zorder value below which artists will be rasterized.
`Axes.set_rasterization_zorder`	Set the zorder threshold for rasterization for vector graphics output.
`Axes.get_window_extent`	Return the Axes bounding box in display space; *args* and *kwargs* are empty.
`Axes.get_tightbbox`	Return the tight bounding box of the axes, including axis and their decorators (xlabel, title, etc).

Projection

Methods used by `Axis` that must be overridden for non-rectilinear Axes.

`Axes.name`	
`Axes.get_xaxis_transform`	Get the transformation used for drawing x-axis labels, ticks and gridlines.
`Axes.get_yaxis_transform`	Get the transformation used for drawing y-axis labels, ticks and gridlines.
`Axes.get_data_ratio`	Return the aspect ratio of the scaled data.
`Axes.get_xaxis_text1_transform`	**Returns:**
`Axes.get_xaxis_text2_transform`	**Returns:**
`Axes.get_yaxis_text1_transform`	**Returns:**
`Axes.get_yaxis_text2_transform`	**Returns:**

Other

`Axes.zorder`	

`Axes.get_default_bbox_extra_artists`	Return a default list of artists that are used for the bounding box calculation.
`Axes.get_transformed_clip_path_and_affine`	Return the clip path with the non-affine part of its transformation applied, and the remaining affine part of its transformation.
`Axes.has_data`	Return whether any artists have been added to the Axes.
`Axes.set`	Set multiple properties at once.