matplotlib.axes.Axes.legend

Axes.legend(*args, **kwargs) [source]

Place a legend on the Axes.

Call signatures:

```
legend()
legend(handles, labels)
legend(handles=handles)
legend(labels)
```

The call signatures correspond to the following different ways to use this method:

1. Automatic detection of elements to be shown in the legend

The elements to be added to the legend are automatically determined, when you do not pass in any extra arguments.

In this case, the labels are taken from the artist. You can specify them either at artist creation or by calling the **set_label()** method on the artist:

```
ax.plot([1, 2, 3], label='Inline label')
ax.legend()
```

or:

```
line, = ax.plot([1, 2, 3])
line.set_label('Label via method')
ax.legend()
```

Specific lines can be excluded from the automatic legend element selection by defining a label starting with an underscore. This is default for all artists, so calling **Axes.legend** without any arguments and without setting the labels manually will result in no legend being drawn.

2. Explicitly listing the artists and labels in the legend

For full control of which artists have a legend entry, it is possible to pass an iterable of legend artists followed by an iterable of legend labels respectively:

```
ax.legend([line1, line2, line3], ['label1', 'label2', 'label3'])
```

3. Explicitly listing the artists in the legend

This is similar to 2, but the labels are taken from the artists' label properties. Example:

```
line1, = ax.plot([1, 2, 3], label='label1')
line2, = ax.plot([1, 2, 3], label='label2')
ax.legend(handles=[line1, line2])
```

4. Labeling existing plot elements

> **Discouraged**
>
> This call signature is discouraged, because the relation between plot elements and labels is only implicit by their order and can easily be mixed up.

To make a legend for all artists on an Axes, call this function with an iterable of strings, one for each legend item. For example:

```
ax.plot([1, 2, 3])
ax.plot([5, 6, 7])
ax.legend(['First line', 'Second line'])
```

Parameters:

handles : *sequence of Artist, optional*

A list of Artists (lines, patches) to be added to the legend. Use this together with *labels*, if you need full control on what is shown in the legend and the automatic mechanism described above is not sufficient. The length of handles and labels should be the same in this case. If they are not, they are truncated to the smaller length.

labels : *list of str, optional*

A list of labels to show next to the artists. Use this together with *handles*, if you need full control on what is shown in the legend and the automatic mechanism described above is not sufficient.

Returns:

Legend

Other Parameters: **loc** : ***str or pair of floats, default:*** *rcParams["legend.loc"]* ***(default:*** *'best'****)*** ***('best' for axes, 'upper right' for figures)***

The location of the legend.

The strings `'upper left', 'upper right', 'lower left', 'lower right'` place the legend at the corresponding corner of the axes/figure.

The strings `'upper center', 'lower center', 'center left', 'center right'` place the legend at the center of the corresponding edge of the axes/figure.

The string `'center'` places the legend at the center of the axes/figure.

The string `'best'` places the legend at the location, among the nine locations defined so far, with the minimum overlap with other drawn artists. This option can be quite slow for plots with large amounts of data; your plotting speed may benefit from providing a specific location.

The location can also be a 2-tuple giving the coordinates of the lower-left corner of the legend in axes coordinates (in which case *bbox_to_anchor* will be ignored).

For back-compatibility, `'center right'` (but no other location) can also be spelled `'right'`, and each "string" locations can also be given as a numeric value:

Location String	Location Code
'best'	0
'upper right'	1
'upper left'	2
'lower left'	3
'lower right'	4
'right'	5
'center left'	6
'center right'	7
'lower center'	8
'upper center'	9
'center'	10

bbox_to_anchor : ***BboxBase, 2-tuple, or 4-tuple of floats***

Box that is used to position the legend in conjunction with *loc*. Defaults to `axes.bbox` (if called as a method to `Axes.legend`) or `figure.bbox` (if `Figure.legend`). This argument allows arbitrary placement of the legend.

Bbox coordinates are interpreted in the coordinate system given by *bbox_transform*, with the default transform Axes or Figure coordinates, depending on which `legend` is called.

If a 4-tuple or `BboxBase` is given, then it specifies the bbox `(x, y, width, height)` that the legend is placed in. To put the legend in the best location in the bottom right quadrant of the axes (or figure):

```
loc='best', bbox_to_anchor=(0.5, 0., 0.5, 0.5)
```

A 2-tuple `(x, y)` places the corner of the legend specified by *loc* at x, y. For example, to put the legend's upper right-hand corner in the center of the axes (or figure) the following keywords can be used:

```
loc='upper right', bbox_to_anchor=(0.5, 0.5)
```

ncol : *int, default: 1*

The number of columns that the legend has.

prop : *None or matplotlib.font_manager.FontProperties or dict*

The font properties of the legend. If None (default), the current `matplotlib.rcParams` will be used.

fontsize : *int or {'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large', 'xx-large'}*

The font size of the legend. If the value is numeric the size will be the absolute font size in points. String values are relative to the current default font size. This argument is only used if *prop* is not specified.

labelcolor : *str or list, default: rcParams["legend.labelcolor"] (default: 'None')*

The color of the text in the legend. Either a valid color string (for example, 'red'), or a list of color strings. The labelcolor can also be made to match the color of the line or marker using 'linecolor', 'markerfacecolor' (or 'mfc'), or 'markeredgecolor' (or 'mec').

Labelcolor can be set globally using `rcParams["legend.labelcolor"]` (default: `'None'`). If None, use `rcParams["text.color"]` (default: `'black'`).

numpoints : *int, default: rcParams["legend.numpoints"] (default: 1)*

The number of marker points in the legend when creating a legend entry for a `Line2D` (line).

scatterpoints : *int, default: rcParams["legend.scatterpoints"] (default: 1)*

The number of marker points in the legend when creating a legend entry for a `PathCollection` (scatter plot).

scatteryoffsets : *iterable of floats, default: [0.375, 0.5, 0.3125]*

The vertical offset (relative to the font size) for the markers created for a scatter plot legend entry. 0.0 is at the base the legend text, and 1.0 is at the top. To draw all markers at the same height, set to `[0.5]`.

markerscale : *float, default: rcParams["legend.markerscale"] (default: 1.0)*

The relative size of legend markers compared with the originally drawn ones.

markerfirst : *bool, default: True*

If *True*, legend marker is placed to the left of the legend label. If *False*, legend marker is placed to the right of the legend label.

frameon : *bool, default: rcParams["legend.frameon"] (default: True)*

Whether the legend should be drawn on a patch (frame).

The legend's background patch edge color. If `"inherit"`, use take `rcParams["axes.edgecolor"]` (default: `'black'`).

mode : *{"expand", None}*

If *mode* is set to `"expand"` the legend will be horizontally expanded to fill the axes area (or *bbox_to_anchor* if defines the legend's size).

bbox_transform : *None or matplotlib.transforms.Transform*

The transform for the bounding box (*bbox_to_anchor*). For a value of `None` (default) the Axes' `transAxes` transform will be used.

title : *str or None*

The legend's title. Default is no title (`None`).

title_fontproperties : *None or matplotlib.font_manager.FontProperties or dict*

The font properties of the legend's title. If None (default), the *title_fontsize* argument will be used if present; if *title_fontsize* is also None, the current `rcParams["legend.title_fontsize"]` (default: `None`) will be used.

title_fontsize : *int or {'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large', 'xx-large'}*

The font size of the legend's title. Note: This cannot be combined with *title_fontproperties*. If you want to set the fontsize alongside other font properties, use the *size* parameter in *title_fontproperties*.

borderpad : *float, default: rcParams["legend.borderpad"] (default: 0.4)*

The fractional whitespace inside the legend border, in font-size units.

labelspacing : *float, default: rcParams["legend.labelspacing"] (default:*

See also

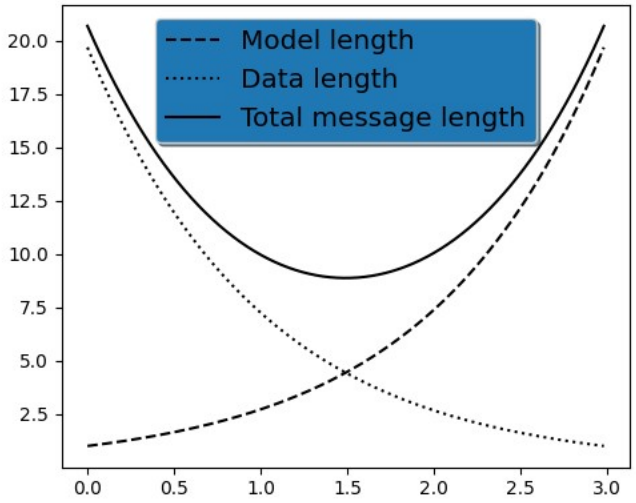
`Figure.legend`

Notes

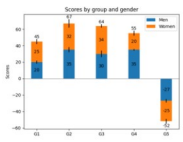
Some artists are not supported by this function. See Legend guide for details.

Examples

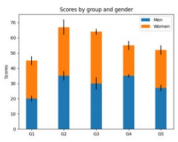
(Source code, png, pdf)



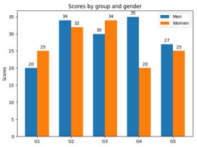
Examples using `matplotlib.axes.Axes.legend`



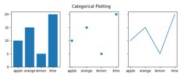
Bar Label Demo



Stacked bar chart

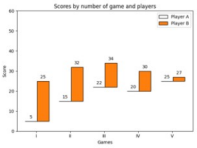


Grouped bar chart with labels

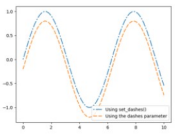


Plotting categorical variables

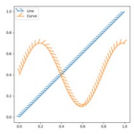
Fill Between and Alpha



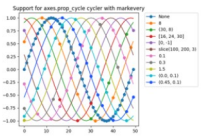
Hat graph



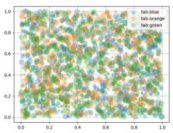
Customizing dashed line styles



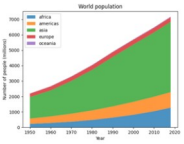
Lines with a ticked patheffect



prop_cycle property markevery in rcParams

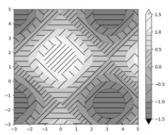


Scatter plots with a legend

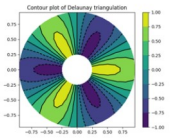


Stackplots and streamgraphs

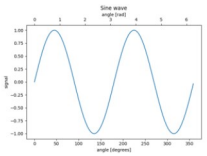
Stairs Demo



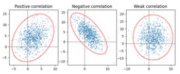
Contourf Hatching



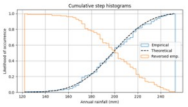
Tricontour Demo



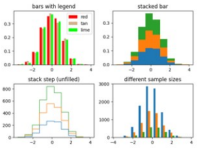
Secondary Axis



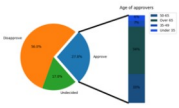
Plot a confidence ellipse of a two-dimensional dataset



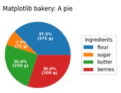
Using histograms to plot a cumulative distribution



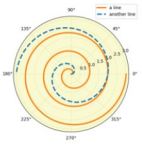
The histogram (hist) function with multiple data sets



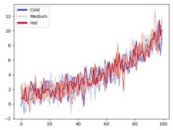
Bar of pie



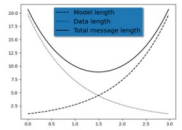
Labeling a pie and a donut



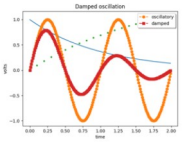
Polar Legend



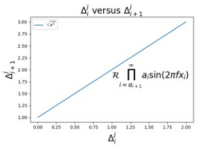
Composing Custom Legends



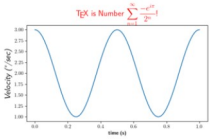
Legend using pre-defined labels



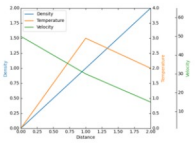
Legend Demo



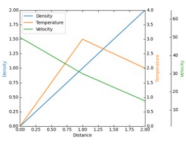
Mathtext



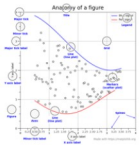
Rendering math equations using TeX



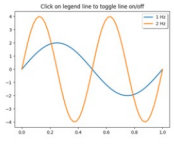
Parasite Axes demo



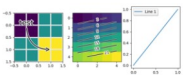
Parasite axis demo



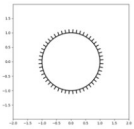
Anatomy of a figure



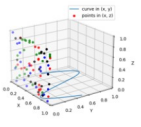
Legend Picking



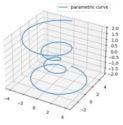
Patheffect Demo



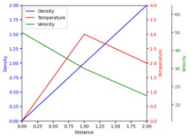
TickedStroke patheffect



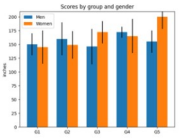
Plot 2D data on 3D plot



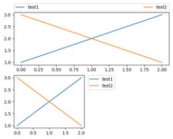
Parametric Curve



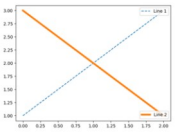
Multiple Yaxis With Spines



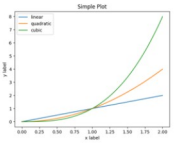
Group barchart with units



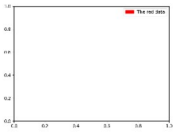
Simple Legend01



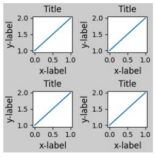
Simple Legend02



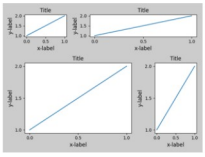
Basic Usage



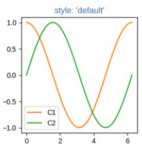
Legend guide



Constrained Layout Guide



Tight Layout guide



Specifying Colors