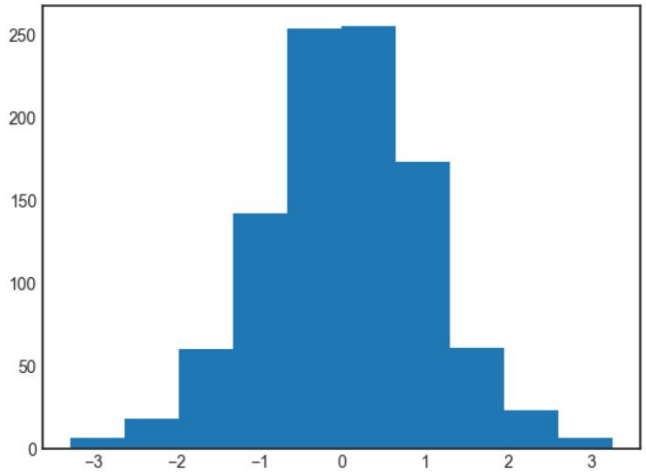
Histogram

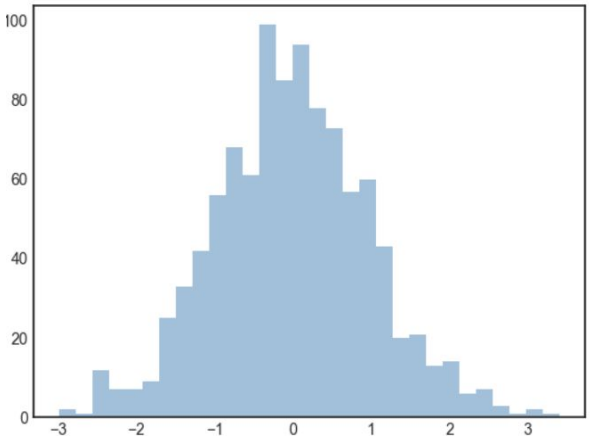
1. Simple histogram




```
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('seaborn-white')

data = np.random.randn(1000)
plt.hist(data)
plt.show()
```

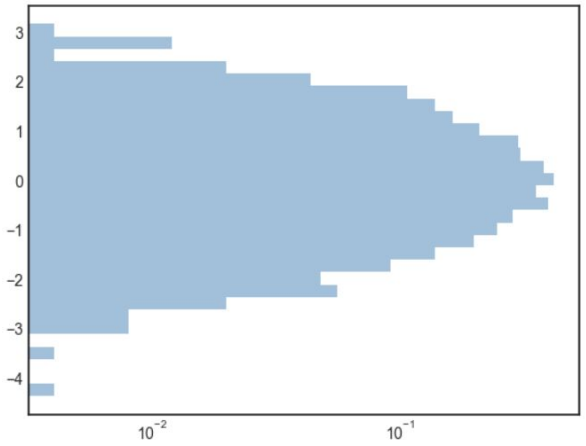
2. Hist_2 (adding some function parameters)




```
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('seaborn-white')

data = np.random.randn(1000)
plt.hist(data, bins=30, histtype='stepfilled',
         color='steelblue', alpha=0.5,
         edgecolor='none');
plt.show()
```

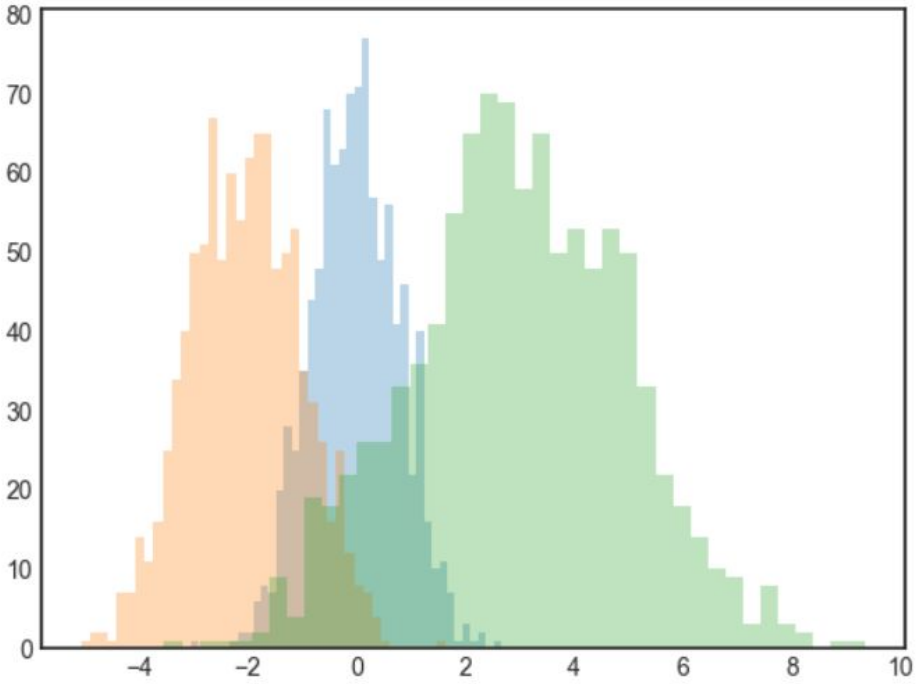
3. Hist_3




```
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('seaborn-white')

data = np.random.randn(1000)
plt.hist(data, bins=30, density=True,
         orientation='horizontal',
         log=True,
         color='steelblue', alpha=0.5,
         edgecolor='none');
plt.show()
```

4. Hist_4




```
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('seaborn-white')

x1 = np.random.normal(0, 0.8, 1000)
x2 = np.random.normal(-2, 1, 1000)
x3 = np.random.normal(3, 2, 1000)

kwargs = dict(histtype='stepfilled',
              alpha=0.3, bins=40)
plt.hist(x1, **kwargs)
plt.hist(x2, **kwargs)
plt.hist(x3, **kwargs);

plt.show()
```

Default parameters:

```
x, bins=None, range=None, density: bool = False, weights=None,
cumulative: bool = False, bottom=None, histtype: str = 'bar',
align: str = 'mid', orientation: str = 'vertical',
rwidth=None, log: bool = False, color=None, label=None,
stacked: bool = False, *, data=None, **kwargs
```

Parameters:

x: (n,) array or sequence

Input values take either a single array or a sequence of arrays that are not required to be of the same length.

bins: int or sequence or str, default: `rcParams["hist.bins"]` (default: `10`)

If ***bins* is an integer**, it defines the number of equal-width bins in the range.

If ***bins* is a sequence**, `[1, 2, 3, 4]`

If ***bins* is a string**, it is one of the binning strategies supported by `numpy.histogram_bin_edges`: 'auto', 'fd', 'doane', 'scott', 'stone', 'rice', 'sturges', or 'sqrt'.

alpha: raspunde de intensitatea culorii

range: tuple or None, default: None

The lower and upper range of the bins. Lower and upper outliers are ignored. If not provided, *range* is `(x.min(), x.max())`. The range has no effect if *bins* is a sequence.

If *bins* is a sequence or *range* is specified, autoscaling is based on the specified bin range instead of the range of x.

density: bool, default: False

If `True`, draw and return a probability density: each bin will display the bin's raw count divided by the total number of counts *and the bin width* (`density = counts / (sum(counts) * np.diff(bins))`), so that the area under the histogram integrates to 1 (`np.sum(density * np.diff(bins)) == 1`).

If *stacked* is also `True`, the sum of the histograms is normalized to 1.

weights: (n,) array-like or None, default: None

An array of weights, of the same shape as *x*. Each value in *x* only contributes its associated weight towards the bin count (instead of 1). If *density* is `True`, the weights are normalized, so that the integral of the density over the range remains 1.

This parameter can be used to draw a histogram of data that has already been binned, e.g. using `numpy.histogram` (by treating each bin as a single point with a weight equal to its count)

```
counts, bins = np.histogram(data)
plt.hist(bins[:-1], bins, weights=counts)
```

(or you may alternatively use `bar()`).

cumulative: bool or -1, default: False

If `True`, then a histogram is computed where each bin gives the counts in that bin plus all bins for smaller values. The last bin gives the total number of datapoints.

If *density* is also `True` then the histogram is normalized such that the last bin equals 1.

If *cumulative* is a number less than 0 (e.g., -1), the direction of accumulation is reversed. In this case, if *density* is also `True`, then the histogram is normalized such that the first bin equals 1.

bottom: array-like, scalar, or None, default: None

Location of the bottom of each bin, ie. bins are drawn from `bottom` to `bottom + hist(x, bins)` If a scalar, the bottom of each bin is shifted by the same amount. If an array, each bin is shifted independently and the length of bottom must match the number of bins. If None, defaults to 0.

histtype{'bar', 'barstacked', 'step', 'stepfilled'}, default: 'bar'

The type of histogram to draw.

- 'bar' is a traditional bar-type histogram. If multiple data are given the bars are arranged side by side.
- 'barstacked' is a bar-type histogram where multiple data are stacked on top of each other.
- 'step' generates a lineplot that is by default unfilled.
- 'stepfilled' generates a lineplot that is by default filled.

align{'left', 'mid', 'right'}, default: 'mid'

The horizontal alignment of the histogram bars.

- 'left': bars are centered on the left bin edges.
- 'mid': bars are centered between the bin edges.
- 'right': bars are centered on the right bin edges.

orientation{'vertical', 'horizontal'}, default: 'vertical'

If 'horizontal', `barh` will be used for bar-type histograms and the *bottom* kwarg will be the left edges.

rwidth: float or None, default: None

The relative width of the bars as a fraction of the bin width. If `None`, automatically compute the width.

Ignored if *histtype* is 'step' or 'stepfilled'.

log: bool, default: False

If `True`, the histogram axis will be set to a log scale. If *log* is `True` and *x* is a 1D array, empty bins will be filtered out and only the non-empty `(n, bins, patches)` will be returned.

color: color or array-like of colors or None, default: None

Color or sequence of colors, one per dataset. Default (`None`) uses the standard line color sequence.

label: str or None, default: None

String, or sequence of strings to match multiple datasets. Bar charts yield multiple patches per dataset, but only the first gets the label, so that `legend` will work as expected.

stacked: bool, default: False

If `True`, multiple data are stacked on top of each other If `False` multiple data are arranged side by side if histtype is 'bar' or on top of each other if histtype is 'step'

****kwargs**

`Patch` properties