To produce a stacked bar plot, pass `stacked=True`:

```
In [22]: df2.plot.bar(stacked=True);
```

To get horizontal bar plots, use the `barh` method:

```
In [23]: df2.plot.barh(stacked=True);
```

Histograms

Histograms can be drawn by using the `DataFrame.plot.hist()` and `Series.plot.hist()` methods.

```
In [24]: df4 = pd.DataFrame({'a': np.random.randn(1000) + 1, 'b': np.random.
↪randn(1000),
   ....:                      'c': np.random.randn(1000) - 1}, columns=['a', 'b', 'c'])
   ....:
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-24-3b054428c392> in <module>
----> 1 df4 = pd.DataFrame({'a': np.random.randn(1000) + 1, 'b': np.random.
↪randn(1000),
      2                     'c': np.random.randn(1000) - 1}, columns=['a', 'b', 'c'])

NameError: name 'pd' is not defined

In [25]: plt.figure();

In [26]: df4.plot.hist(alpha=0.5)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-26-d12a7608cec9> in <module>
----> 1 df4.plot.hist(alpha=0.5)
```

(continues on next page)

(continued from previous page)

```
NameError: name 'df4' is not defined
```

A histogram can be stacked using `stacked=True`. Bin size can be changed using the `bins` keyword.

```
In [27]: plt.figure();

In [28]: df4.plot.hist(stacked=True, bins=20)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-28-9a4bef475383> in <module>
----> 1 df4.plot.hist(stacked=True, bins=20)

NameError: name 'df4' is not defined
```

You can pass other keywords supported by matplotlib `hist`. For example, horizontal and cumulative histograms can be drawn by `orientation='horizontal'` and `cumulative=True`.

```
In [29]: plt.figure();

In [30]: df4['a'].plot.hist(orientation='horizontal', cumulative=True)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-30-c49999bfb88a> in <module>
----> 1 df4['a'].plot.hist(orientation='horizontal', cumulative=True)

NameError: name 'df4' is not defined
```

See the `hist` method and the matplotlib hist documentation for more.

The existing interface `DataFrame.hist` to plot histogram still can be used.

```
In [31]: plt.figure();

In [32]: df['A'].diff().hist()
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-32-620f128ae072> in <module>
----> 1 df['A'].diff().hist()

NameError: name 'df' is not defined
```

`DataFrame.hist()` plots the histograms of the columns on multiple subplots:

```
In [33]: plt.figure()
Out[33]: <Figure size 640x480 with 0 Axes>

In [34]: df.diff().hist(color='k', alpha=0.5, bins=50)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-34-742660109dc1> in <module>
----> 1 df.diff().hist(color='k', alpha=0.5, bins=50)

NameError: name 'df' is not defined
```

The `by` keyword can be specified to plot grouped histograms:

```
In [35]: data = pd.Series(np.random.randn(1000))
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-35-cd9ac77fc4c4> in <module>
----> 1 data = pd.Series(np.random.randn(1000))

NameError: name 'pd' is not defined

In [36]: data.hist(by=np.random.randint(0, 4, 1000), figsize=(6, 4))
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-36-9248a2062b4d> in <module>
----> 1 data.hist(by=np.random.randint(0, 4, 1000), figsize=(6, 4))

NameError: name 'data' is not defined
```

Box plots

Boxplot can be drawn calling `Series.plot.box()` and `DataFrame.plot.box()`, or `DataFrame.boxplot()` to visualize the distribution of values within each column.

For instance, here is a boxplot representing five trials of 10 observations of a uniform random variable on [0,1).

```
In [37]: df = pd.DataFrame(np.random.rand(10, 5), columns=['A', 'B', 'C', 'D', 'E'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-37-a2f471686f35> in <module>
----> 1 df = pd.DataFrame(np.random.rand(10, 5), columns=['A', 'B', 'C', 'D', 'E'])

NameError: name 'pd' is not defined

In [38]: df.plot.box()
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-38-8765cf6ed5ce> in <module>
----> 1 df.plot.box()

NameError: name 'df' is not defined
```

Boxplot can be colorized by passing `color` keyword. You can pass a `dict` whose keys are `boxes`, `whiskers`, `medians` and `caps`. If some keys are missing in the `dict`, default colors are used for the corresponding artists. Also, boxplot has `sym` keyword to specify fliers style.

When you pass other type of arguments via `color` keyword, it will be directly passed to matplotlib for all the `boxes`, `whiskers`, `medians` and `caps` colorization.

The colors are applied to every boxes to be drawn. If you want more complicated colorization, you can get each drawn artists by passing *return_type*.

```
In [39]: color = {'boxes': 'DarkGreen', 'whiskers': 'DarkOrange',
   ....:          'medians': 'DarkBlue', 'caps': 'Gray'}
   ....:

In [40]: df.plot.box(color=color, sym='r+')
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-40-2a54c0d52eaf> in <module>
----> 1 df.plot.box(color=color, sym='r+')

NameError: name 'df' is not defined
```

Also, you can pass other keywords supported by matplotlib `boxplot`. For example, horizontal and custom-positioned boxplot can be drawn by `vert=False` and `positions` keywords.

```
In [41]: df.plot.box(vert=False, positions=[1, 4, 5, 6, 8])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-41-6b82106697f4> in <module>
----> 1 df.plot.box(vert=False, positions=[1, 4, 5, 6, 8])

NameError: name 'df' is not defined
```

See the `boxplot` method and the matplotlib boxplot documentation for more.

The existing interface `DataFrame.boxplot` to plot boxplot still can be used.

```
In [42]: df = pd.DataFrame(np.random.rand(10, 5))
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-42-fd2300c25153> in <module>
----> 1 df = pd.DataFrame(np.random.rand(10, 5))

NameError: name 'pd' is not defined

In [43]: plt.figure();

In [44]: bp = df.boxplot()
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-44-5b6d837d4b1a> in <module>
----> 1 bp = df.boxplot()

NameError: name 'df' is not defined
```

You can create a stratified boxplot using the `by` keyword argument to create groupings. For instance,

```
In [45]: df = pd.DataFrame(np.random.rand(10, 2), columns=['Col1', 'Col2'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-45-da722611cdbb> in <module>
----> 1 df = pd.DataFrame(np.random.rand(10, 2), columns=['Col1', 'Col2'])

NameError: name 'pd' is not defined

In [46]: df['X'] = pd.Series(['A', 'A', 'A', 'A', 'A', 'B', 'B', 'B', 'B', 'B'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-46-b2bbda782b40> in <module>
----> 1 df['X'] = pd.Series(['A', 'A', 'A', 'A', 'A', 'B', 'B', 'B', 'B', 'B'])

NameError: name 'pd' is not defined

In [47]: plt.figure();

In [48]: bp = df.boxplot(by='X')
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-48-8598e842a6ba> in <module>
----> 1 bp = df.boxplot(by='X')
```

(continues on next page)

(continued from previous page)

```
NameError: name 'df' is not defined
```

You can also pass a subset of columns to plot, as well as group by multiple columns:

```
In [49]: df = pd.DataFrame(np.random.rand(10, 3), columns=['Col1', 'Col2', 'Col3'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-49-9128a4c3d9c4> in <module>
----> 1 df = pd.DataFrame(np.random.rand(10, 3), columns=['Col1', 'Col2', 'Col3'])

NameError: name 'pd' is not defined

In [50]: df['X'] = pd.Series(['A', 'A', 'A', 'A', 'A', 'B', 'B', 'B', 'B', 'B'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-50-b2bbda782b40> in <module>
----> 1 df['X'] = pd.Series(['A', 'A', 'A', 'A', 'A', 'B', 'B', 'B', 'B', 'B'])

NameError: name 'pd' is not defined

In [51]: df['Y'] = pd.Series(['A', 'B', 'A', 'B', 'A', 'B', 'A', 'B', 'A', 'B'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
```

(continues on next page)

(continued from previous page)

```
<ipython-input-51-9bb56ebffdc5> in <module>
----> 1 df['Y'] = pd.Series(['A', 'B', 'A', 'B', 'A', 'B', 'A', 'B', 'A', 'B'])

NameError: name 'pd' is not defined

In [52]: plt.figure();

In [53]: bp = df.boxplot(column=['Col1', 'Col2'], by=['X', 'Y'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-53-e2989456fa84> in <module>
----> 1 bp = df.boxplot(column=['Col1', 'Col2'], by=['X', 'Y'])

NameError: name 'df' is not defined
```

In `boxplot`, the return type can be controlled by the `return_type`, keyword. The valid choices are `{"axes", "dict", "both", None}`. Faceting, created by `DataFrame.boxplot` with the `by` keyword, will affect the output type as well:

return_type=	Faceted	Output type
None	No	axes
None	Yes	2-D ndarray of axes
'axes'	No	axes
'axes'	Yes	Series of axes
'dict'	No	dict of artists
'dict'	Yes	Series of dicts of artists
'both'	No	namedtuple
'both'	Yes	Series of namedtuples

`Groupby.boxplot` always returns a `Series` of `return_type`.

```
In [54]: np.random.seed(1234)

In [55]: df_box = pd.DataFrame(np.random.randn(50, 2))
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-55-043b0e16e969> in <module>
----> 1 df_box = pd.DataFrame(np.random.randn(50, 2))

NameError: name 'pd' is not defined

In [56]: df_box['g'] = np.random.choice(['A', 'B'], size=50)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-56-e39101f788cc> in <module>
----> 1 df_box['g'] = np.random.choice(['A', 'B'], size=50)

NameError: name 'df_box' is not defined

In [57]: df_box.loc[df_box['g'] == 'B', 1] += 3
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-57-996ee2e7f114> in <module>
----> 1 df_box.loc[df_box['g'] == 'B', 1] += 3

NameError: name 'df_box' is not defined

In [58]: bp = df_box.boxplot(by='g')
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-58-8fc769e009a9> in <module>
----> 1 bp = df_box.boxplot(by='g')

NameError: name 'df_box' is not defined
```

The subplots above are split by the numeric columns first, then the value of the g column. Below the subplots are first split by the value of g, then by the numeric columns.

```
In [59]: bp = df_box.groupby('g').boxplot()
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-59-900c71ff9ec1> in <module>
----> 1 bp = df_box.groupby('g').boxplot()

NameError: name 'df_box' is not defined
```

Area plot

You can create area plots with `Series.plot.area()` and `DataFrame.plot.area()`. Area plots are stacked by default. To produce stacked area plot, each column must be either all positive or all negative values.

When input data contains *NaN*, it will be automatically filled by 0. If you want to drop or fill by different values, use `dataframe.dropna()` or `dataframe.fillna()` before calling *plot*.

```
In [60]: df = pd.DataFrame(np.random.rand(10, 4), columns=['a', 'b', 'c', 'd'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-60-1599b5508584> in <module>
----> 1 df = pd.DataFrame(np.random.rand(10, 4), columns=['a', 'b', 'c', 'd'])

NameError: name 'pd' is not defined

In [61]: df.plot.area();
```

To produce an unstacked plot, pass `stacked=False`. Alpha value is set to 0.5 unless otherwise specified:

```
In [62]: df.plot.area(stacked=False);
```

Scatter plot

Scatter plot can be drawn by using the `DataFrame.plot.scatter()` method. Scatter plot requires numeric columns for the x and y axes. These can be specified by the `x` and `y` keywords.

```
In [63]: df = pd.DataFrame(np.random.rand(50, 4), columns=['a', 'b', 'c', 'd'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-63-a5ddb87a0bbe> in <module>
----> 1 df = pd.DataFrame(np.random.rand(50, 4), columns=['a', 'b', 'c', 'd'])

NameError: name 'pd' is not defined

In [64]: df.plot.scatter(x='a', y='b');
```

To plot multiple column groups in a single axes, repeat `plot` method specifying target `ax`. It is recommended to specify `color` and `label` keywords to distinguish each groups.

```
In [65]: ax = df.plot.scatter(x='a', y='b', color='DarkBlue', label='Group 1');

In [66]: df.plot.scatter(x='c', y='d', color='DarkGreen', label='Group 2', ax=ax);
```

The keyword `c` may be given as the name of a column to provide colors for each point:

```
In [67]: df.plot.scatter(x='a', y='b', c='c', s=50);
```

You can pass other keywords supported by matplotlib `scatter`. The example below shows a bubble chart using a column of the `DataFrame` as the bubble size.

```
In [68]: df.plot.scatter(x='a', y='b', s=df['c'] * 200);
```

See the `scatter` method and the matplotlib scatter documentation for more.

Hexagonal bin plot

You can create hexagonal bin plots with `DataFrame.plot.hexbin()`. Hexbin plots can be a useful alternative to scatter plots if your data are too dense to plot each point individually.

```
In [69]: df = pd.DataFrame(np.random.randn(1000, 2), columns=['a', 'b'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-69-e243cb9efde5> in <module>
----> 1 df = pd.DataFrame(np.random.randn(1000, 2), columns=['a', 'b'])

NameError: name 'pd' is not defined

In [70]: df['b'] = df['b'] + np.arange(1000)
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-70-09ef1c00dd7f> in <module>
----> 1 df['b'] = df['b'] + np.arange(1000)

NameError: name 'df' is not defined

In [71]: df.plot.hexbin(x='a', y='b', gridsize=25)
```

(continues on next page)

(continued from previous page)

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-71-48fcf967aa91> in <module>
----> 1 df.plot.hexbin(x='a', y='b', gridsize=25)

NameError: name 'df' is not defined
```

A useful keyword argument is `gridsize`; it controls the number of hexagons in the x-direction, and defaults to 100. A larger `gridsize` means more, smaller bins.

By default, a histogram of the counts around each `(x, y)` point is computed. You can specify alternative aggregations by passing values to the `C` and `reduce_C_function` arguments. `C` specifies the value at each `(x, y)` point and `reduce_C_function` is a function of one argument that reduces all the values in a bin to a single number (e.g. `mean`, `max`, `sum`, `std`). In this example the positions are given by columns `a` and `b`, while the value is given by column `z`. The bins are aggregated with NumPy's `max` function.

```
In [72]: df = pd.DataFrame(np.random.randn(1000, 2), columns=['a', 'b'])
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-72-e243cb9efde5> in <module>
----> 1 df = pd.DataFrame(np.random.randn(1000, 2), columns=['a', 'b'])

NameError: name 'pd' is not defined
```

(continues on next page)