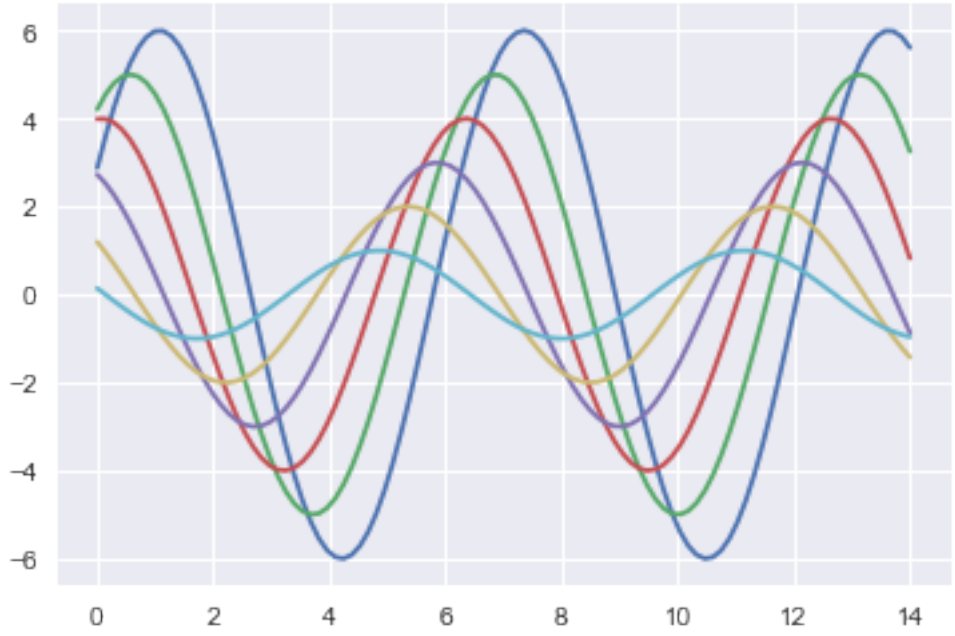
???seaborn

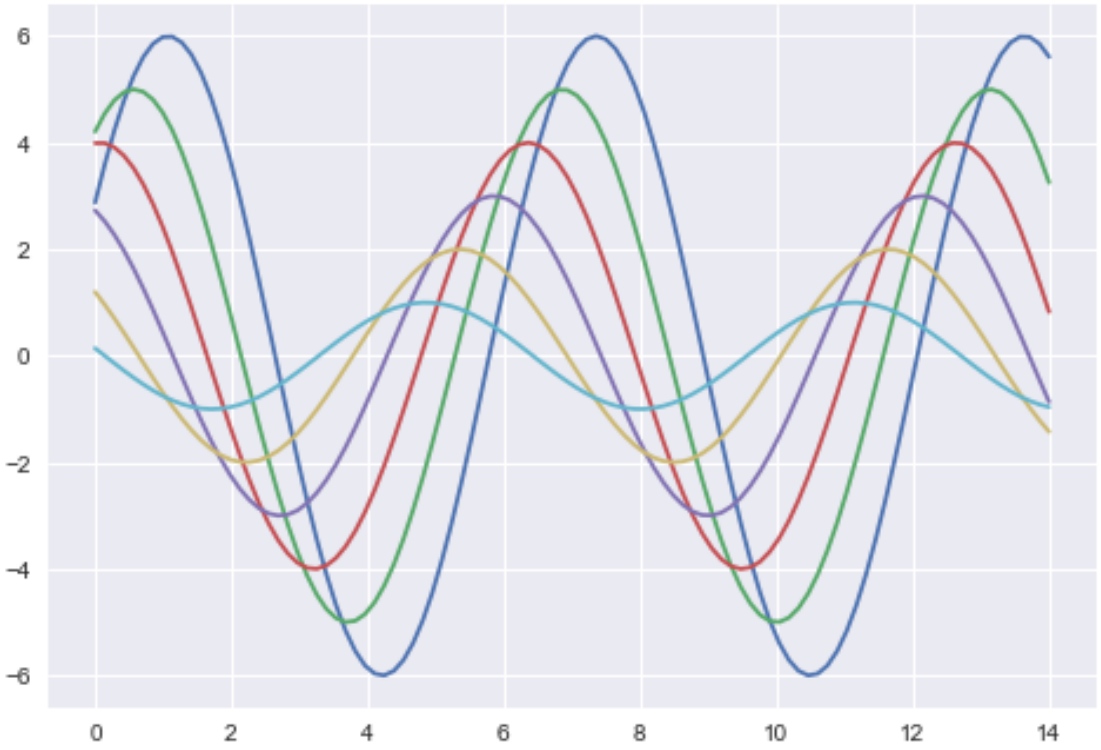
2018 年 2 月 9 日

In [1]:
```
import seaborn as sns
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
def sinplot(flip=1):
    x = np.linspace(0, 14, 100)
    for i in range(1, 7):
        plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)

sinplot()
```



```
In [3]: sns.set()
        sinplot()
```

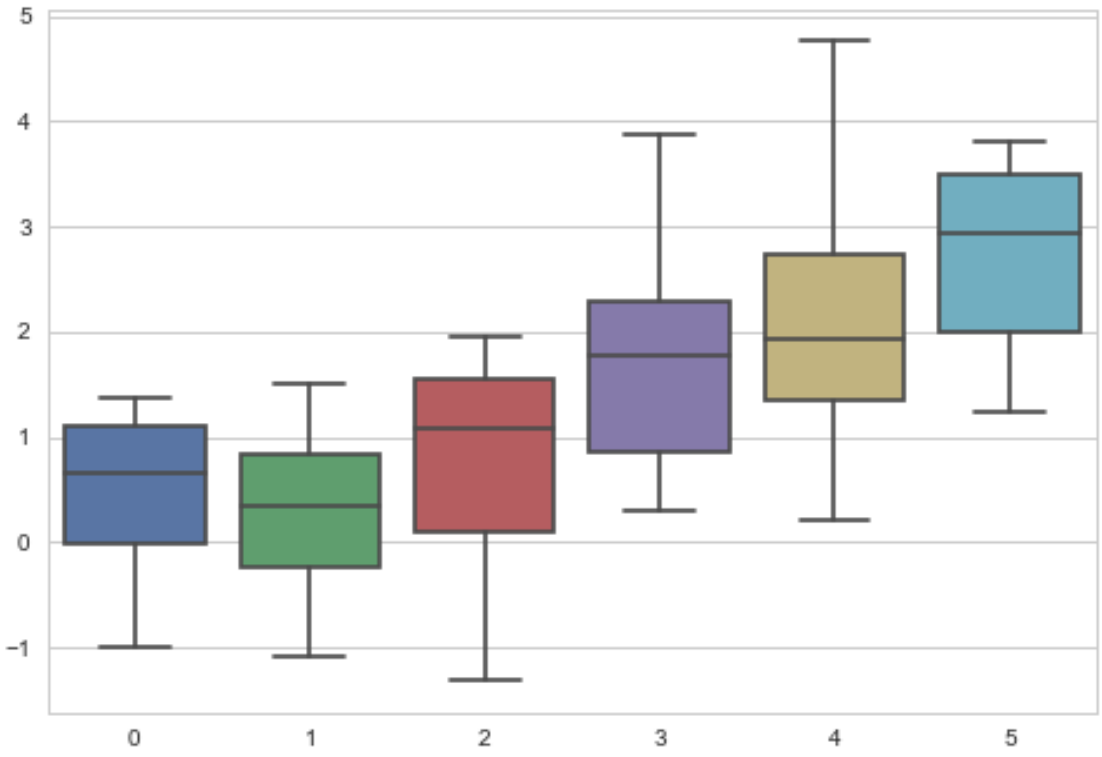


5 种主题风格

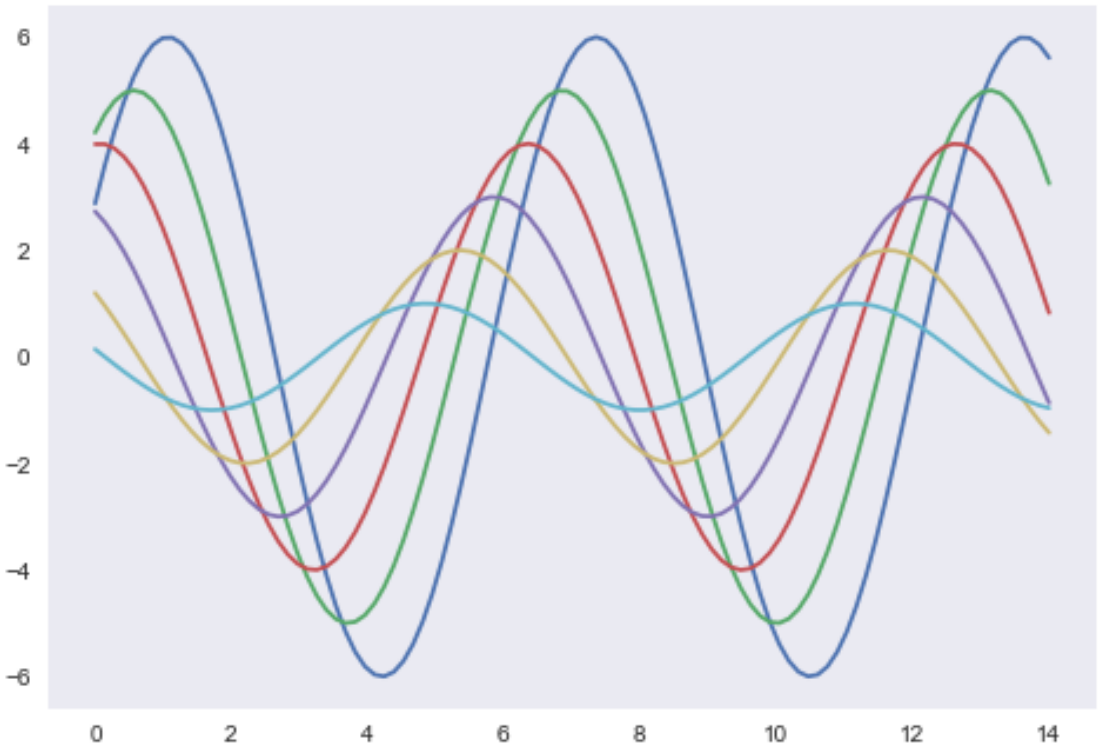
```
darkgrid
whitegrid
dark
white
ticks
```

```
In [4]: sns.set_style("whitegrid")
        data = np.random.normal(size=(20, 6)) + np.arange(6) / 2
        sns.boxplot(data=data)
```

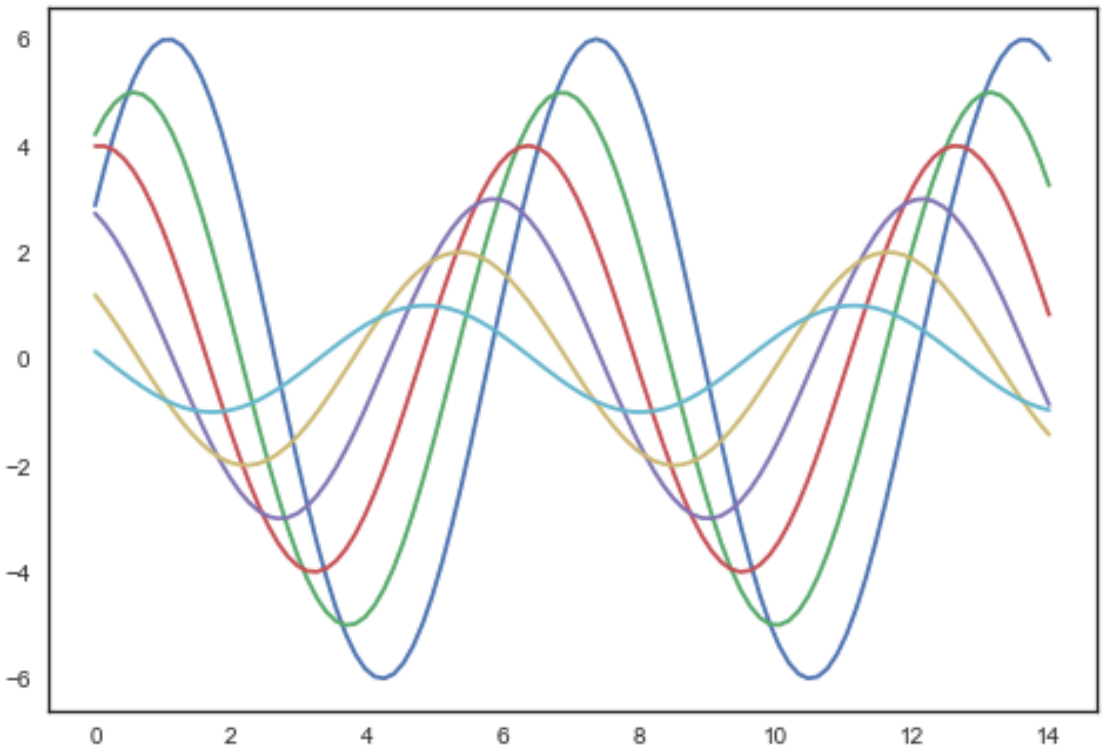
```
Out[4]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6c85af2b0>
```



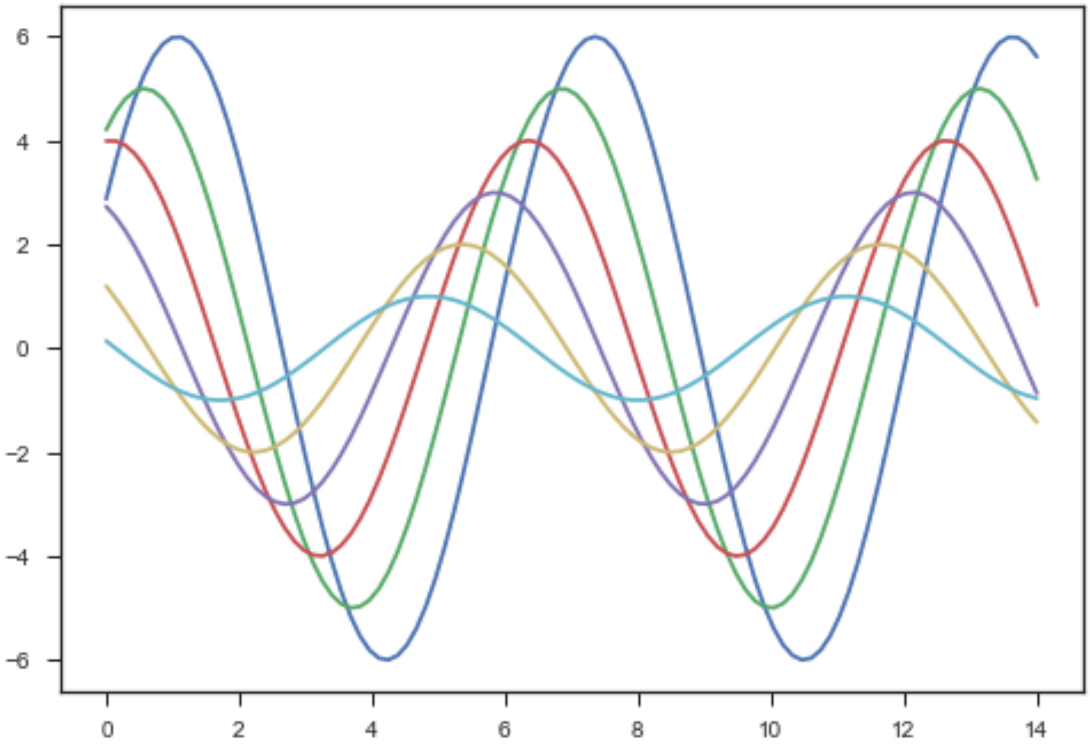
In [5]:
```
sns.set_style("dark")
sinplot()
```



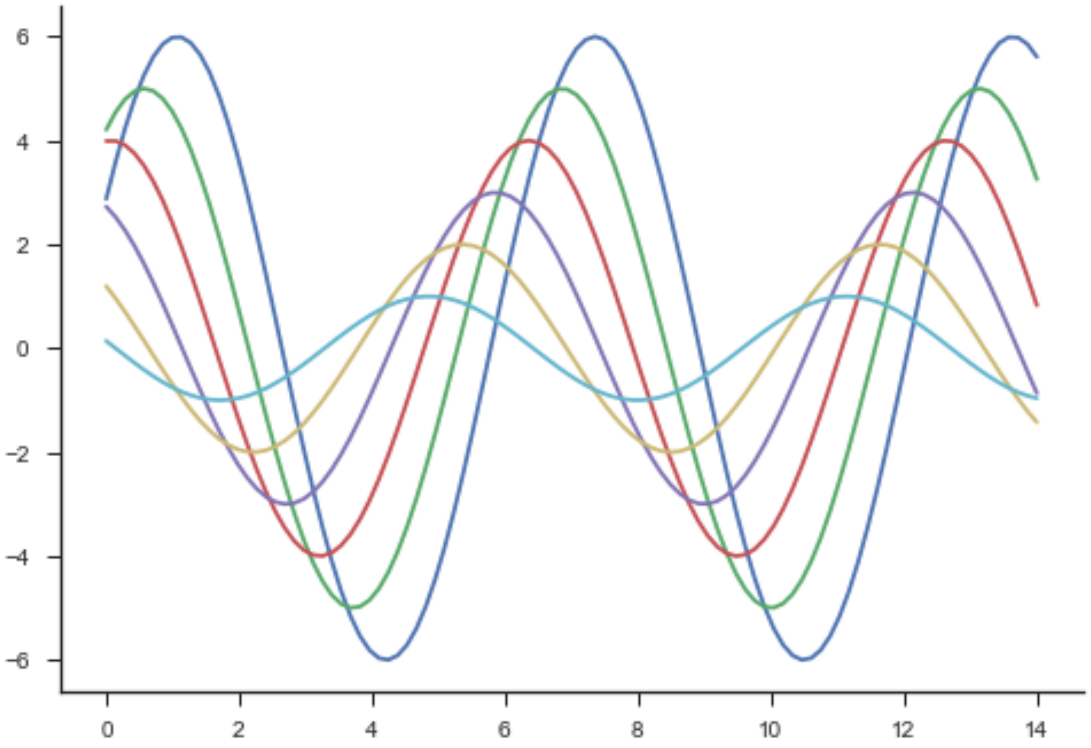
In [6]:
```
sns.set_style("white")
sinplot()
```



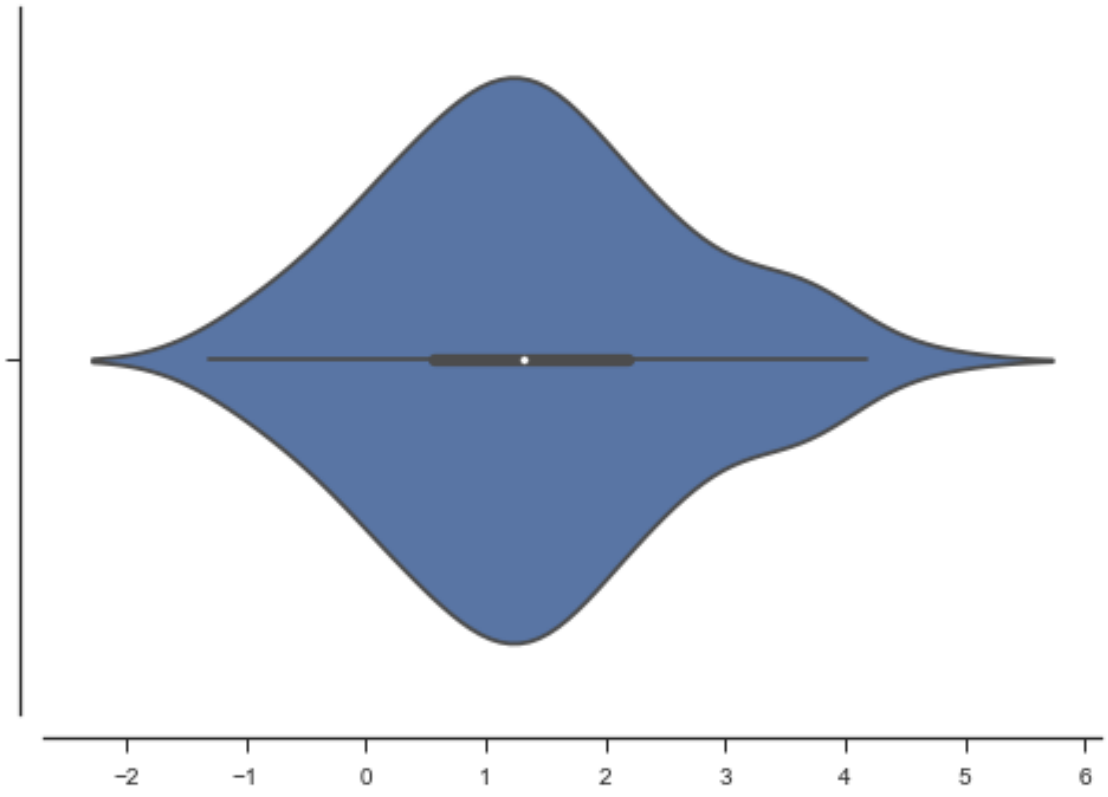

```
In [7]: sns.set_style("ticks")
        sinplot()
```



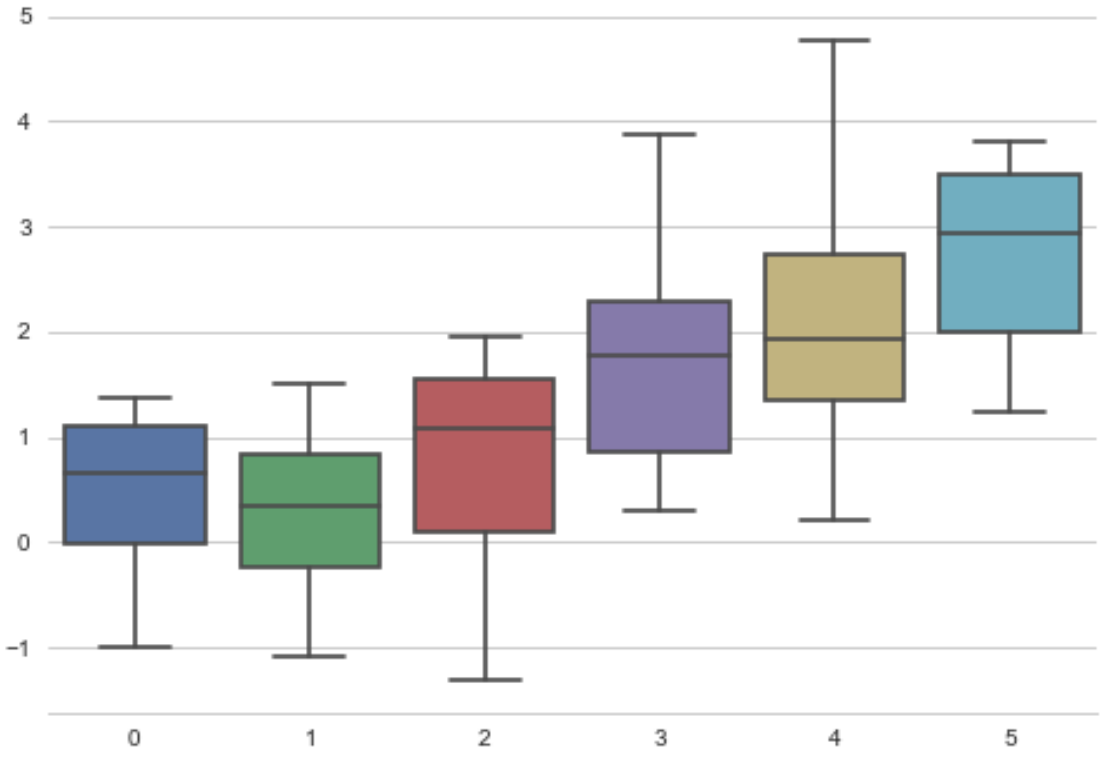
In [8]:
```
# 上边和右边的边框隐藏了
sinplot()
sns.despine()
```



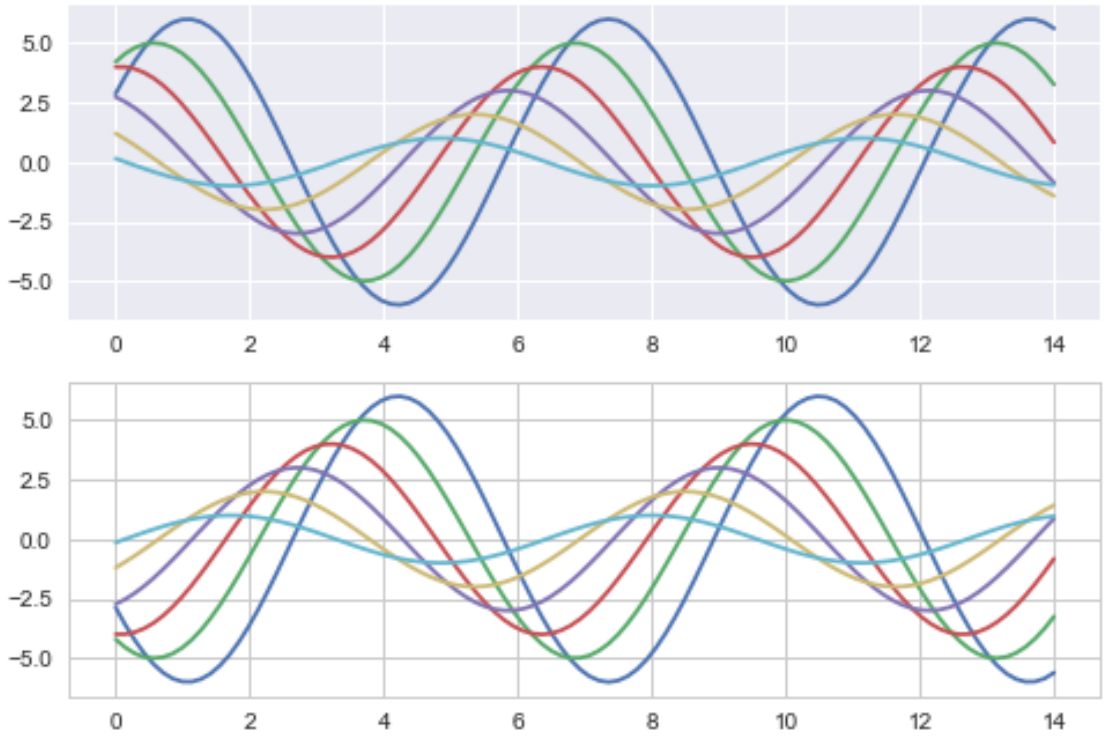
In [9]:
```
#f, ax = plt.subplots()
sns.violinplot(data)
sns.despine(offset=10)#offset=10 和轴线的距离
```



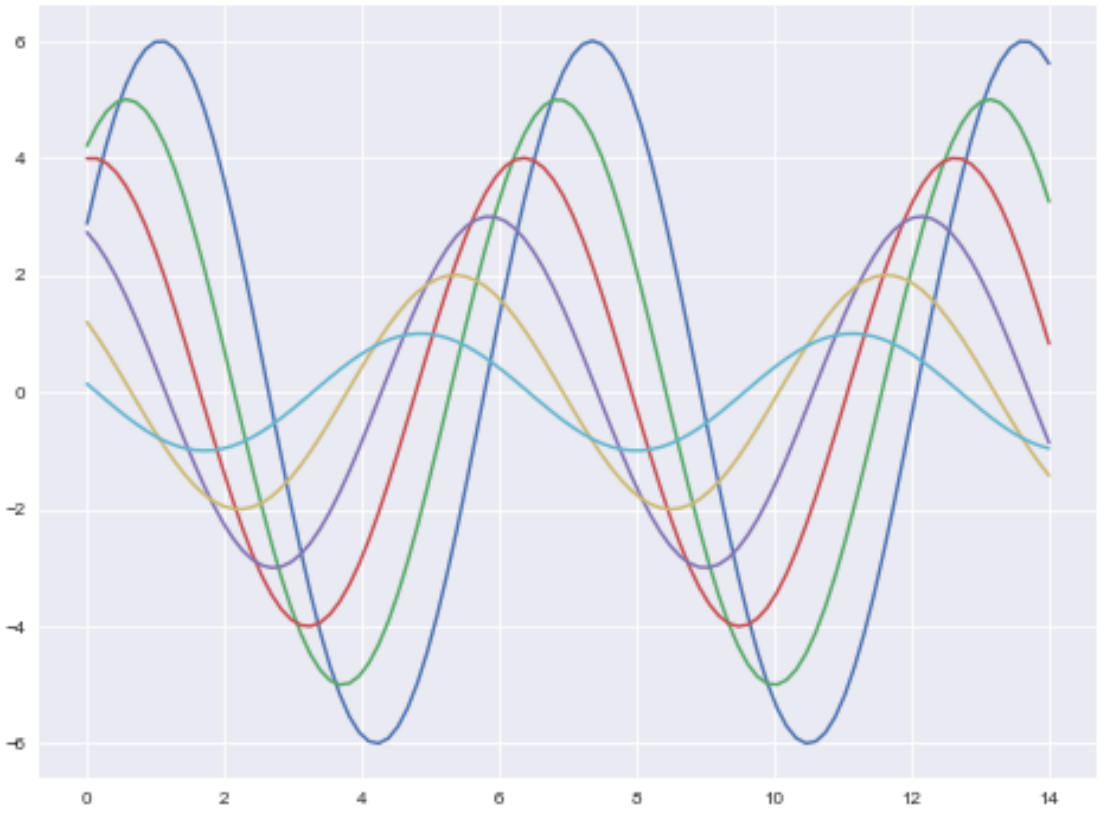

```
In [10]: sns.set_style("whitegrid")
         sns.boxplot(data=data, palette="deep")#palette="deep" 箱线图的线变粗
         sns.despine(left=True)# 隐藏左边的轴
```



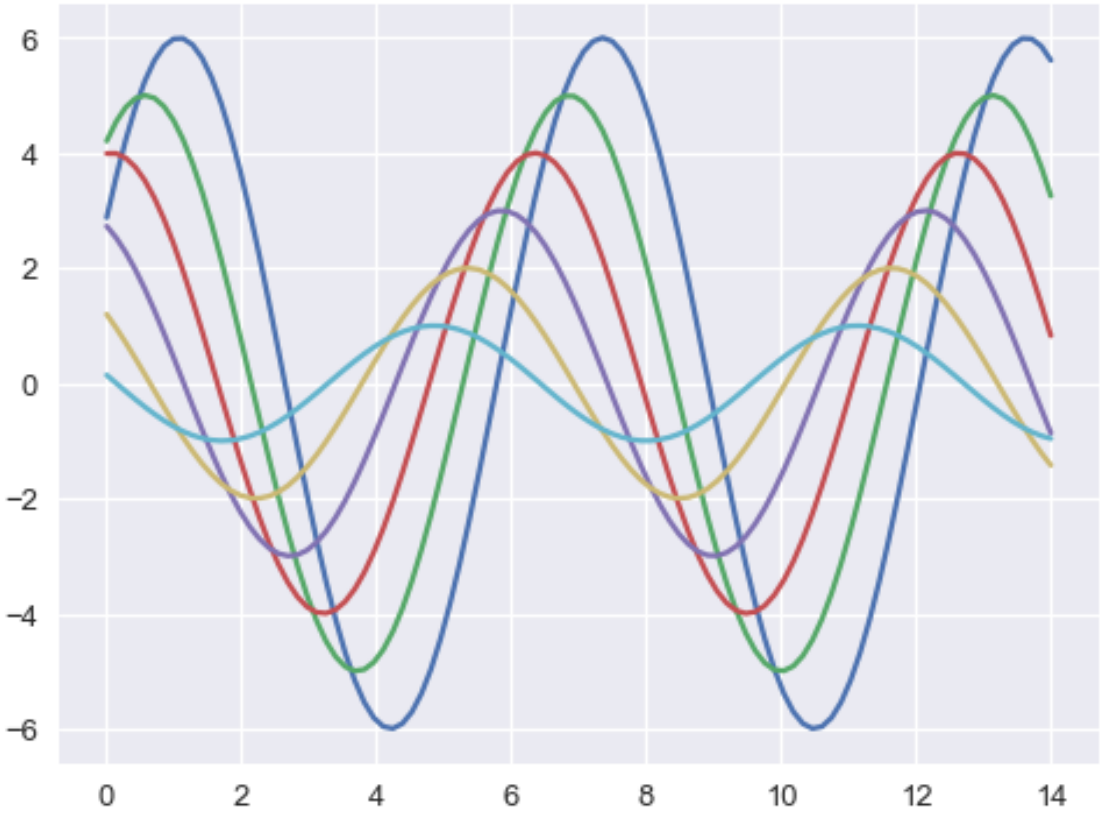

```
In [11]: with sns.axes_style("darkgrid"): #with 里面的风格与其他图形的不同
             plt.subplot(211)
             sinplot()
         plt.subplot(212)
         sinplot(-1)
```



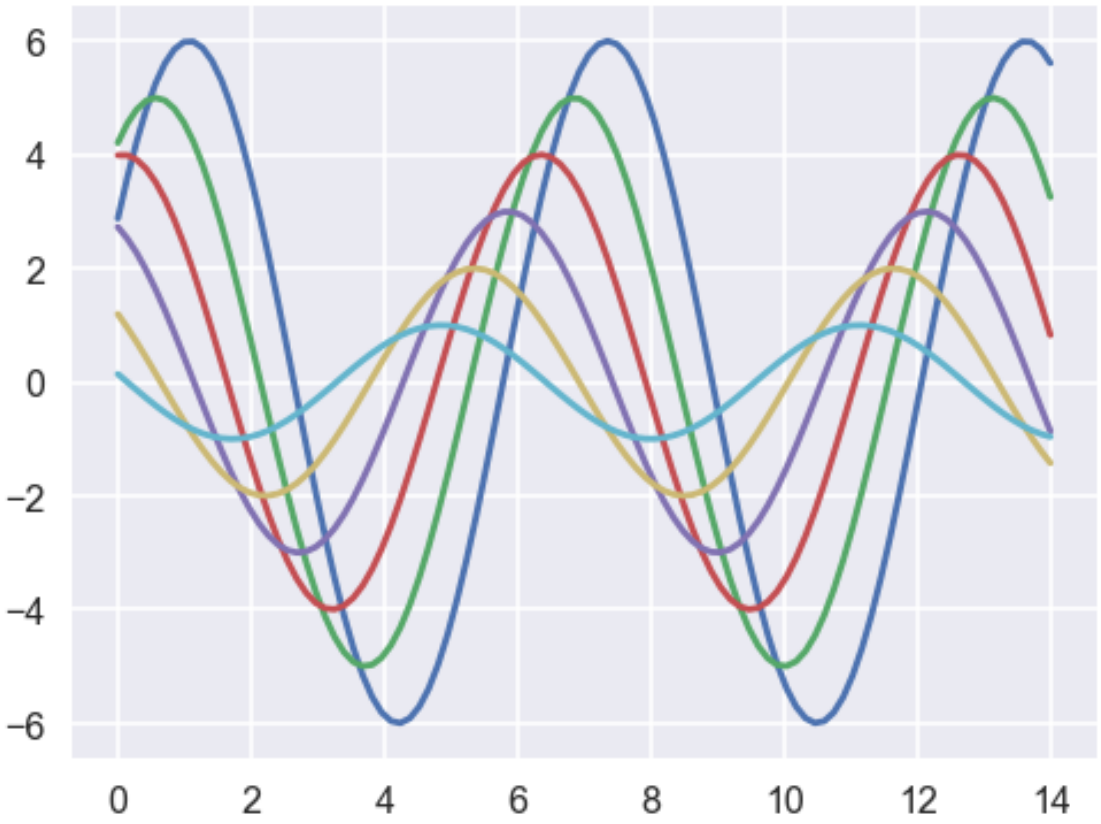

```
In [12]: sns.set()

In [13]: sns.set_context("paper")
         plt.figure(figsize=(8, 6))
         sinplot()
```



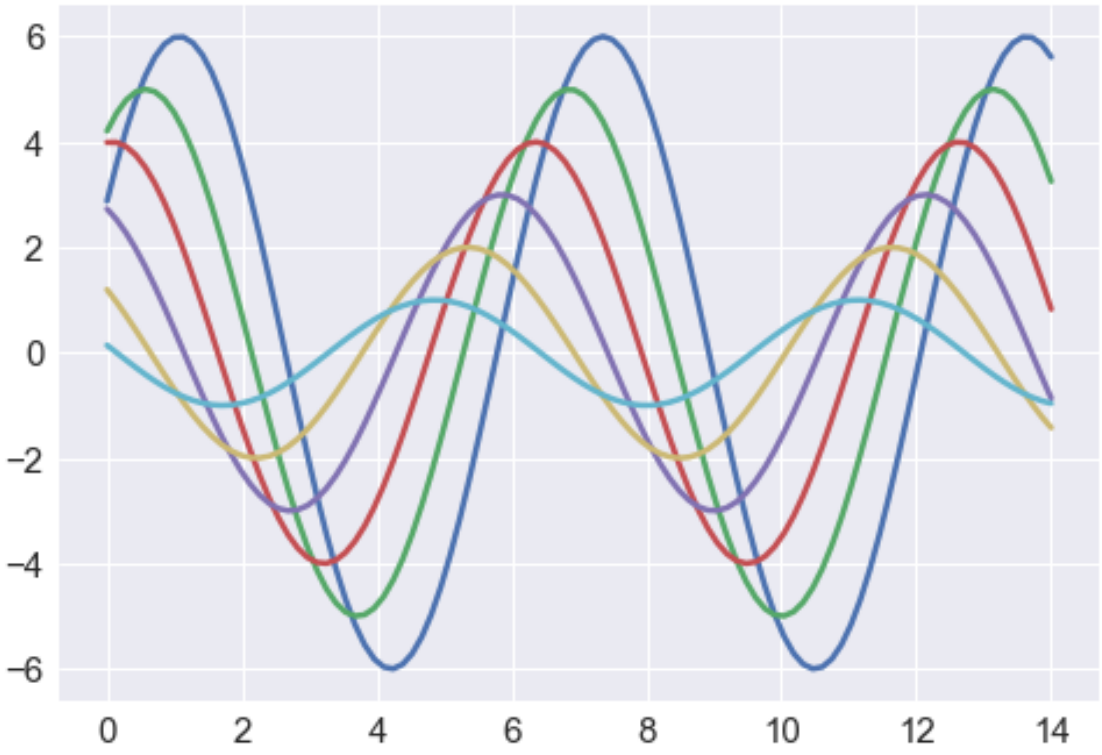

```
In [14]: sns.set_context("talk")
         plt.figure(figsize=(8, 6))
         sinplot()
```




```
In [15]: sns.set_context("poster")
         plt.figure(figsize=(8, 6))
         sinplot()
```




```
In [16]: sns.set_context("notebook", font_scale=1.5, rc={"lines.linewidth": 2.5})
         sinplot()
```




```
In [17]: import numpy as np
         import seaborn as sns
         import matplotlib.pyplot as plt
         %matplotlib inline
         sns.set(rc={"figure.figsize": (6, 6)})
```

1 调色板

- 颜色很重要
- color_palette() 能传入任何 Matplotlib 所支持的颜色
- color_palette() 不写参数则默认颜色
- set_palette() 设置所有图的颜色

1.0.1 分类色板

```
In [18]: current_palette = sns.color_palette()
         sns.palplot(current_palette)
```

6 个默认的颜色循环主题：deep, muted, pastel, bright, dark, colorblind

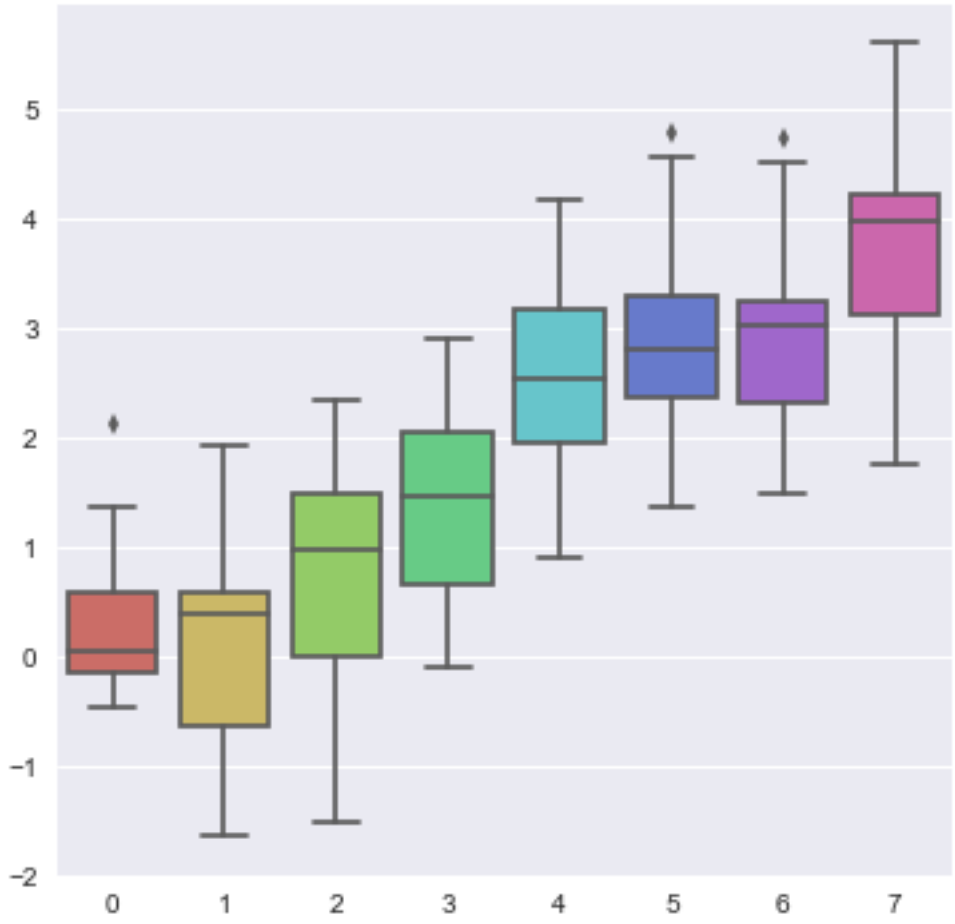
2 圆形画板

当你有六个以上的分类要区分时，最简单的方法就是在一个圆形的颜色空间中画出均匀间隔的颜色 (这样的色调会保持亮度和饱和度不变)。这是大多数的当他们需要使用比当前默认颜色循环中设置的颜色更多时的默认方案。

最常用的方法是使用 hls 的颜色空间，这是 RGB 值的一个简单转换。

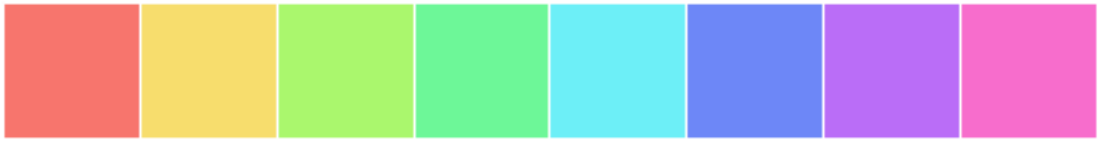
```
In [19]: sns.palplot(sns.color_palette("hls", 8))
```

```
In [20]: data = np.random.normal(size=(20, 8)) + np.arange(8) / 2
         sns.boxplot(data=data,palette=sns.color_palette("hls", 8))

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6c9a97860>
```



hls_palette() 函数来控制颜色的亮度和饱和 * l-亮度 lightness * s-饱和 saturation

```
In [21]: sns.palplot(sns.hls_palette(8, l=.7, s=.9))
```



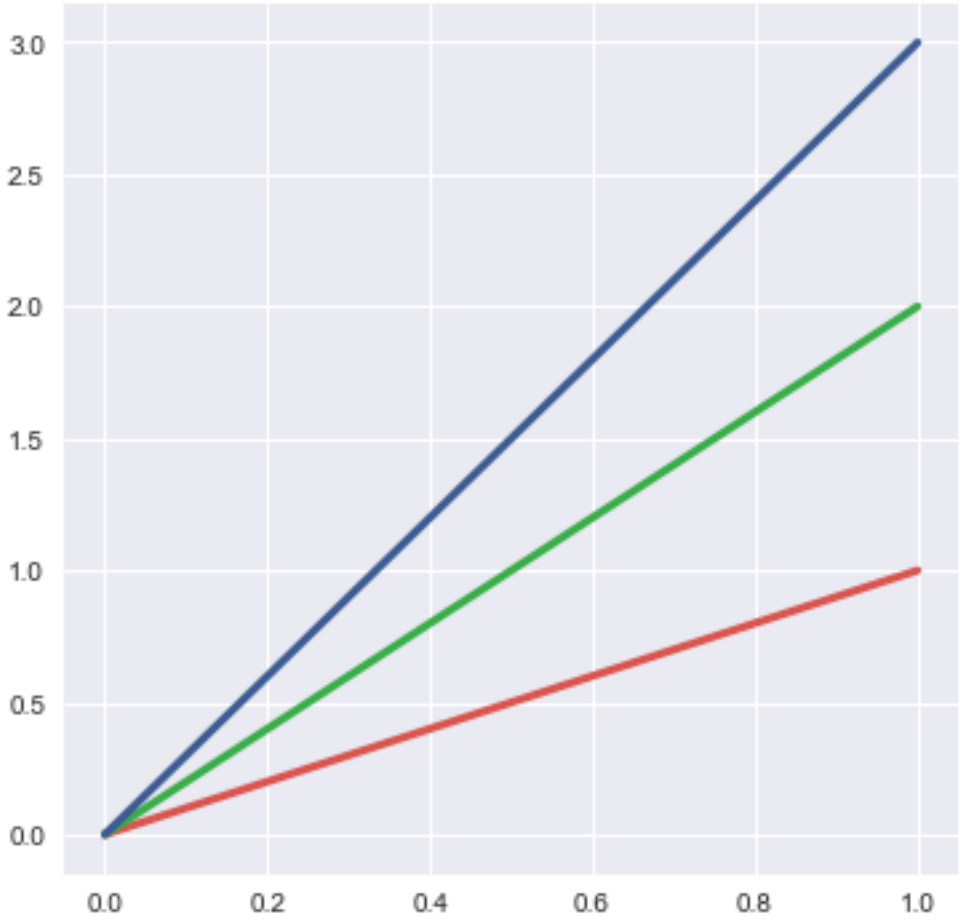
```
In [22]: sns.palplot(sns.color_palette("Paired",8))
```

3 使用 xkcd 颜色来命名颜色

xkcd 包含了一套众包努力的针对随机 RGB 色的命名。产生了 954 个可以随时通过 xdcd_rgb 字典中调用的命名颜色。

```
In [23]: plt.plot([0, 1], [0, 1], sns.xkcd_rgb["pale red"], lw=3)
         plt.plot([0, 1], [0, 2], sns.xkcd_rgb["medium green"], lw=3)
         plt.plot([0, 1], [0, 3], sns.xkcd_rgb["denim blue"], lw=3)
```

```
Out[23]: [<matplotlib.lines.Line2D at 0x1b6c8580ba8>]
```




```
In [24]: colors = ["windows blue", "amber", "greyish", "faded green", "dusty purple"]
         sns.palplot(sns.xkcd_palette(colors))
```

4 连续色板

色彩随数据变换，比如数据越来越重要则颜色越来越深

```
In [25]: sns.palplot(sns.color_palette("Blues"))
```

如果想要翻转渐变，可以在面板名称中添加一个 _r 后缀

```
In [26]: sns.palplot(sns.color_palette("BuGn_r"))
```

5 cubehelix_palette() 调色板

色调线性变换

```
In [27]: sns.palplot(sns.color_palette("cubehelix", 8))
```

In [28]:
```
sns.palplot(sns.cubehelix_palette(8, start=.5, rot=-.75))
```

In [29]:
```
sns.palplot(sns.cubehelix_palette(8, start=.75, rot=-.150))
```

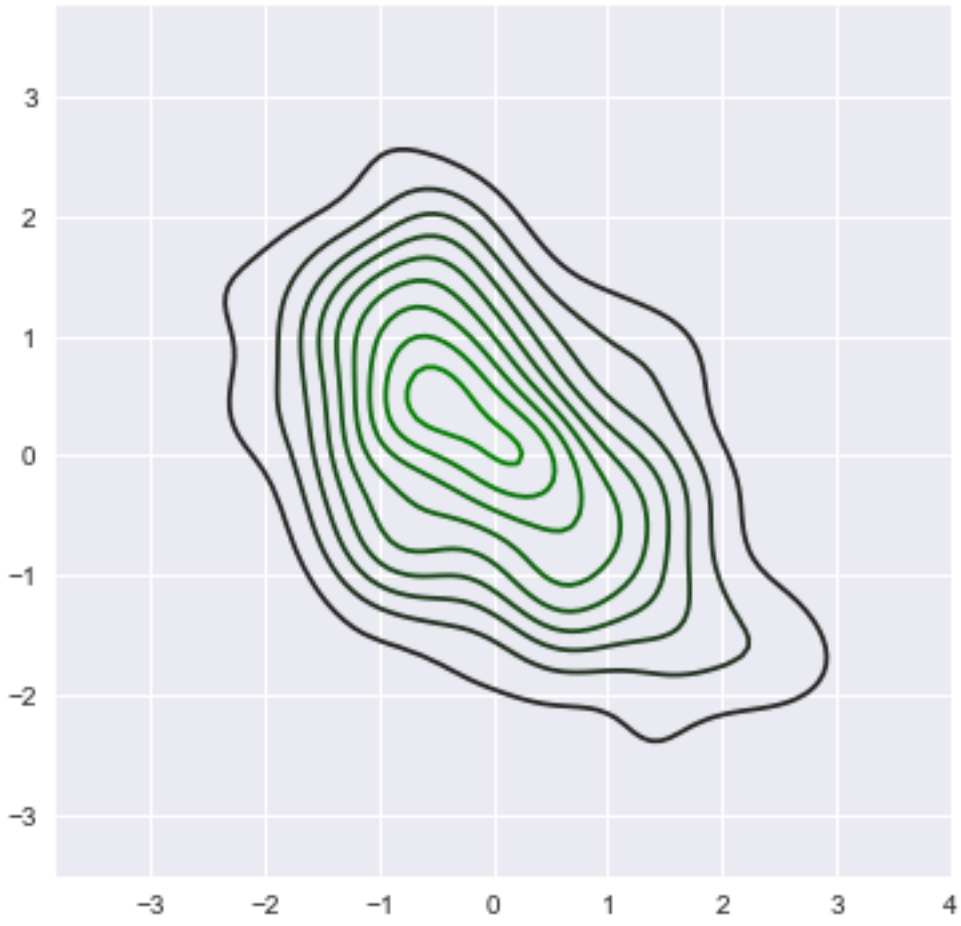
6 light_palette() 和 dark_palette() 调用定制连续调色板

In [30]:
```
sns.palplot(sns.light_palette("green"))
```

In [31]:
```
sns.palplot(sns.dark_palette("purple"))
```

In [32]:
```
sns.palplot(sns.light_palette("navy", reverse=True))
```

In [33]:
```
x, y = np.random.multivariate_normal([0, 0], [[1, -.5], [-.5, 1]], size=300).T
pal = sns.dark_palette("green", as_cmap=True)
sns.kdeplot(x, y, cmap=pal);
```

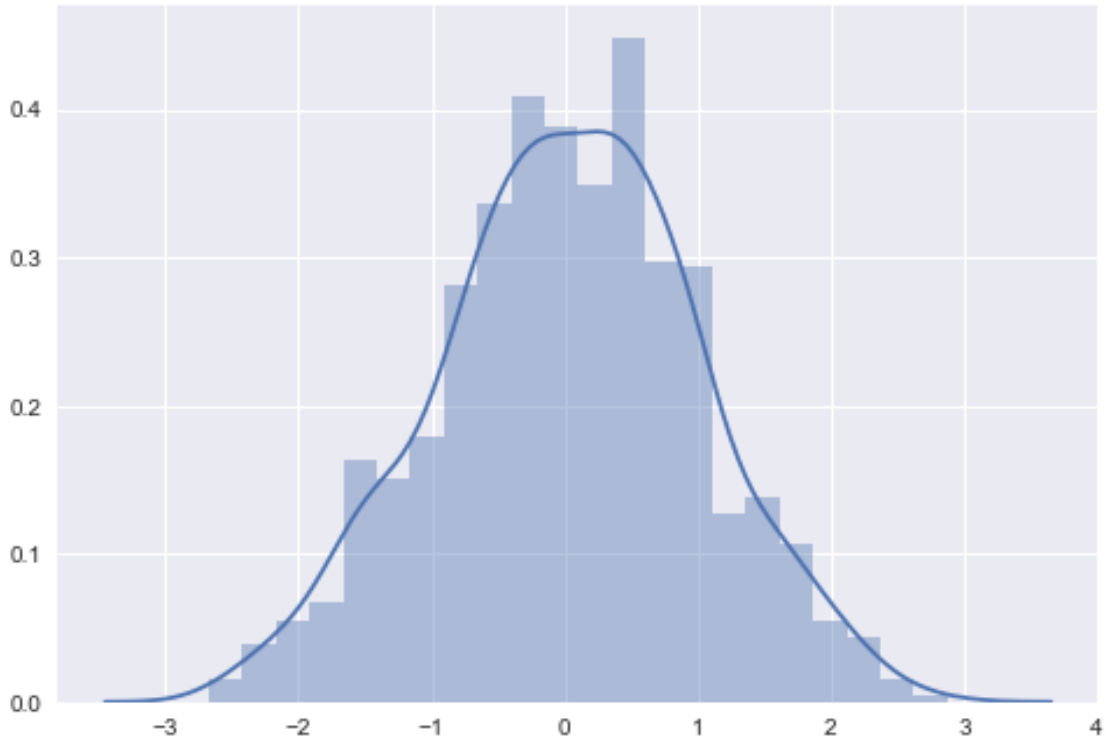


In [34]:
```
sns.palplot(sns.light_palette((210, 90, 60), input="husl"))
```

In [35]:
```
%matplotlib inline
import numpy as np
import pandas as pd
from scipy import stats, integrate
import matplotlib.pyplot as plt

import seaborn as sns
sns.set(color_codes=True)
np.random.seed(sum(map(ord, "distributions")))
```

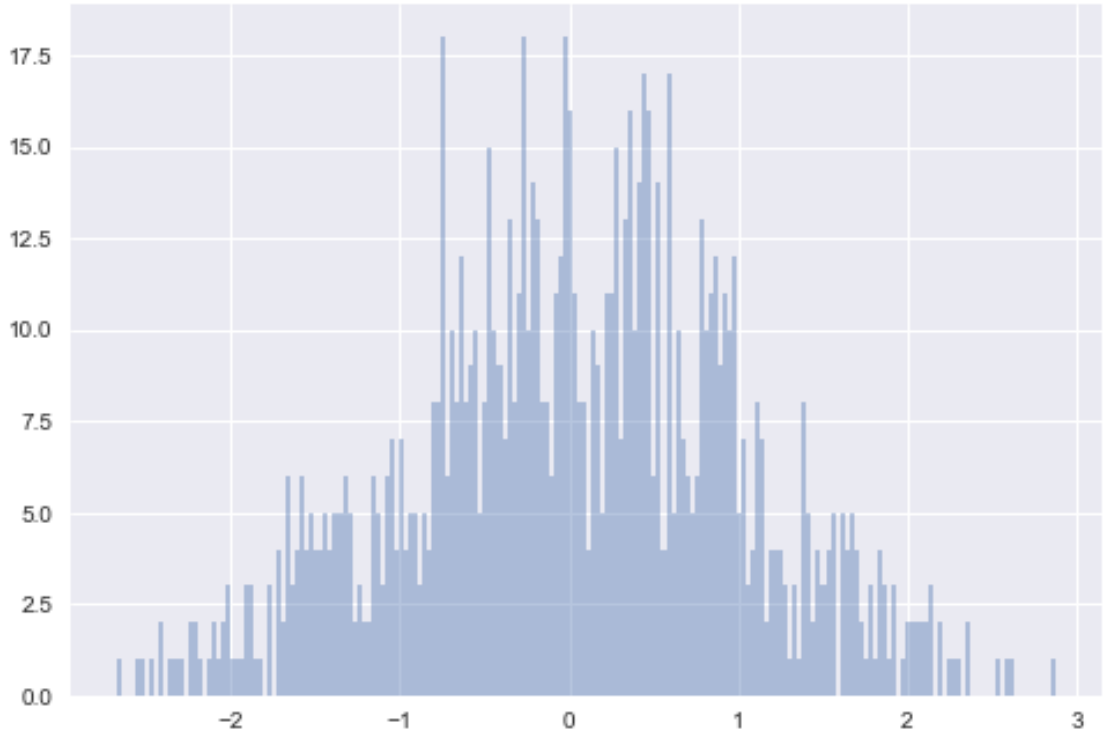
In [36]:
```
x = np.random.normal(size=1000)
sns.distplot(x,kde=True)
```

Out[36]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b6c83e8ef0>
```



```
In [37]: sns.distplot(x, bins=200, kde=False)
```

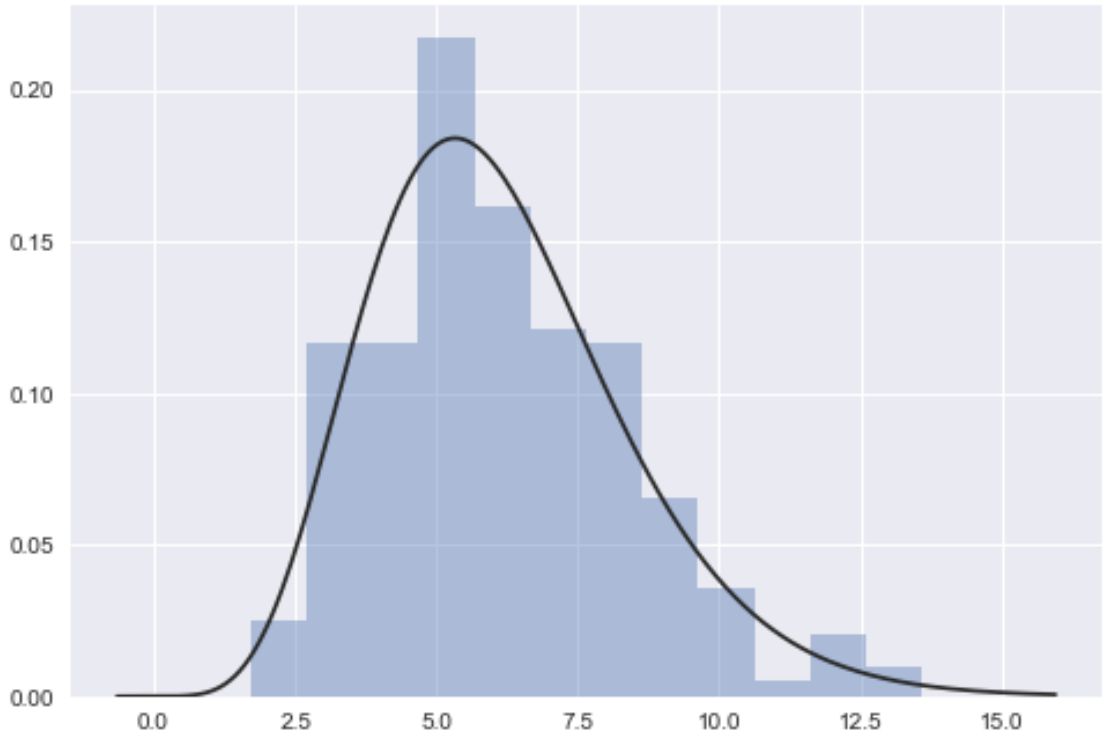
```
Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6c9d27518>
```



6.0.1 数据分布情况

```
In [38]: x = np.random.gamma(6, size=200)
         sns.distplot(x, kde=False, fit=stats.gamma)
```

```
Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6c9d66e80>
```



根据均值和协方差生成数据

```
In [39]: mean, cov = [0, 1], [(1, .5), (.5, 1)]
         data = np.random.multivariate_normal(mean, cov, 200)
         df = pd.DataFrame(data, columns=["x", "y"])
         df
```

Out[39]:

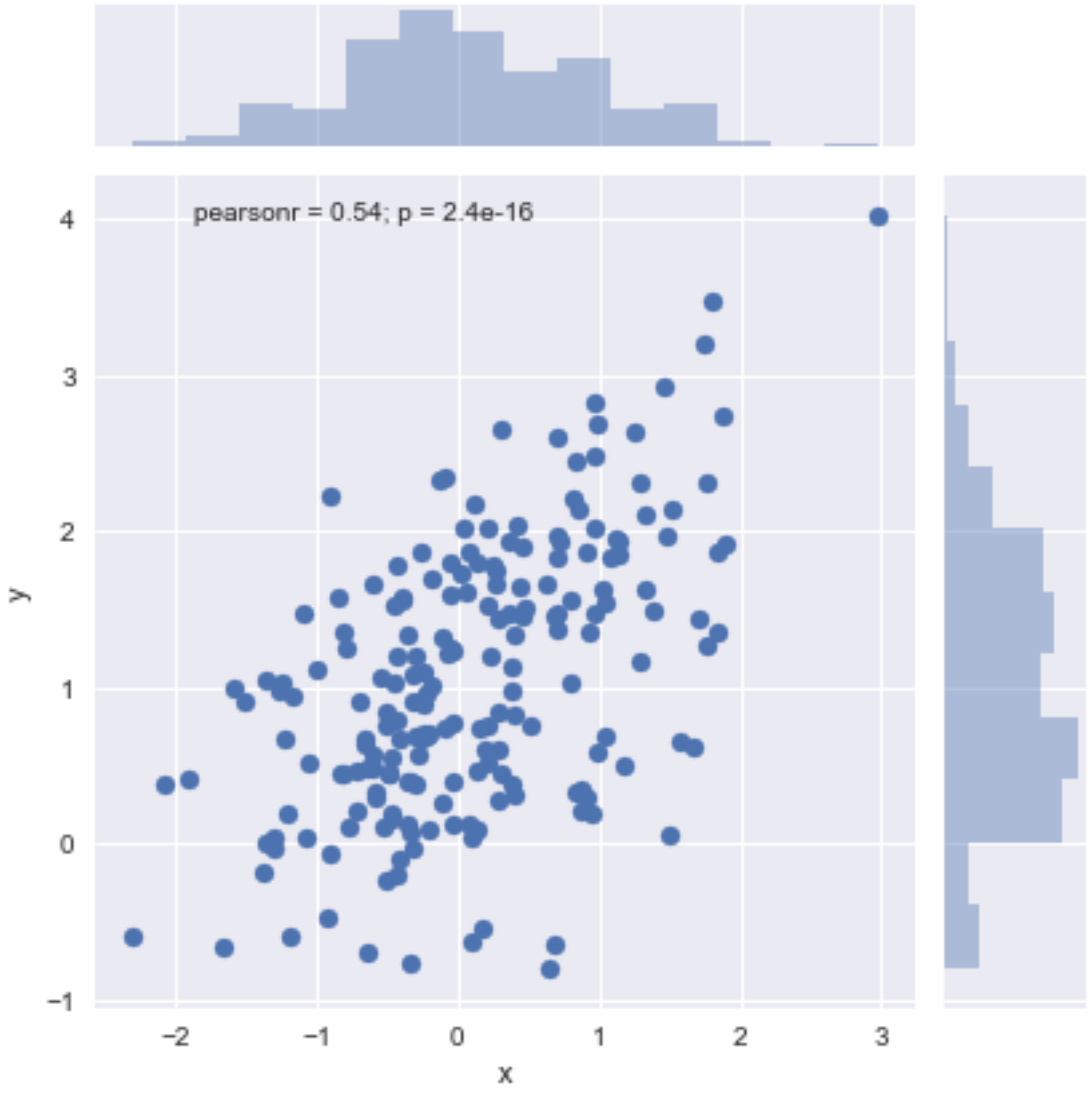
	x	y
0	1.676181	0.617184
1	-0.277674	0.908385
2	-0.459462	1.533641
3	0.694615	1.477441
4	1.833586	1.871051
5	0.255788	1.779415
6	-0.225153	0.696963
7	1.756729	1.261638
8	-0.429231	-0.190816
9	0.294405	0.286579
10	-0.637367	-0.688198
11	0.811442	2.203326

```
12  -0.388126  1.579915
13   1.290688  2.312854
14  -1.069405  0.043629
15  -0.502820  0.844799
16  -0.707670  0.464328
17  -1.093671  1.469160
18   1.323136  2.102123
19   0.930221  1.358323
20   0.306449  2.653722
21   1.392772  1.498629
22  -0.435836  1.773167
23   0.207133  0.752234
24   0.653694 -0.796045
25  -0.996174  1.108219
26  -0.508410 -0.228991
27   1.478623  1.968072
28  -0.050648  1.248472
29   1.883204  2.740215
..        ...       ...
170 -0.469297  0.553858
171 -0.401133  1.561512
172  0.966859  2.480942
173 -0.697959  0.908074
174  0.031451  2.016234
175  0.918416  1.872006
176  0.128707  0.094237
177  0.398856  1.342384
178 -0.211975  0.706119
179  0.205568  1.524847
180 -0.609815  1.663075
181  0.407503  0.314043
182  0.076599  1.861411
183  0.398998  0.819909
184 -1.180396 -0.585879
185 -0.338818 -0.756098
186 -0.808002  0.446369
187 -0.055044  1.798897
188  0.910619  0.288950
189  0.983979  0.589193
```

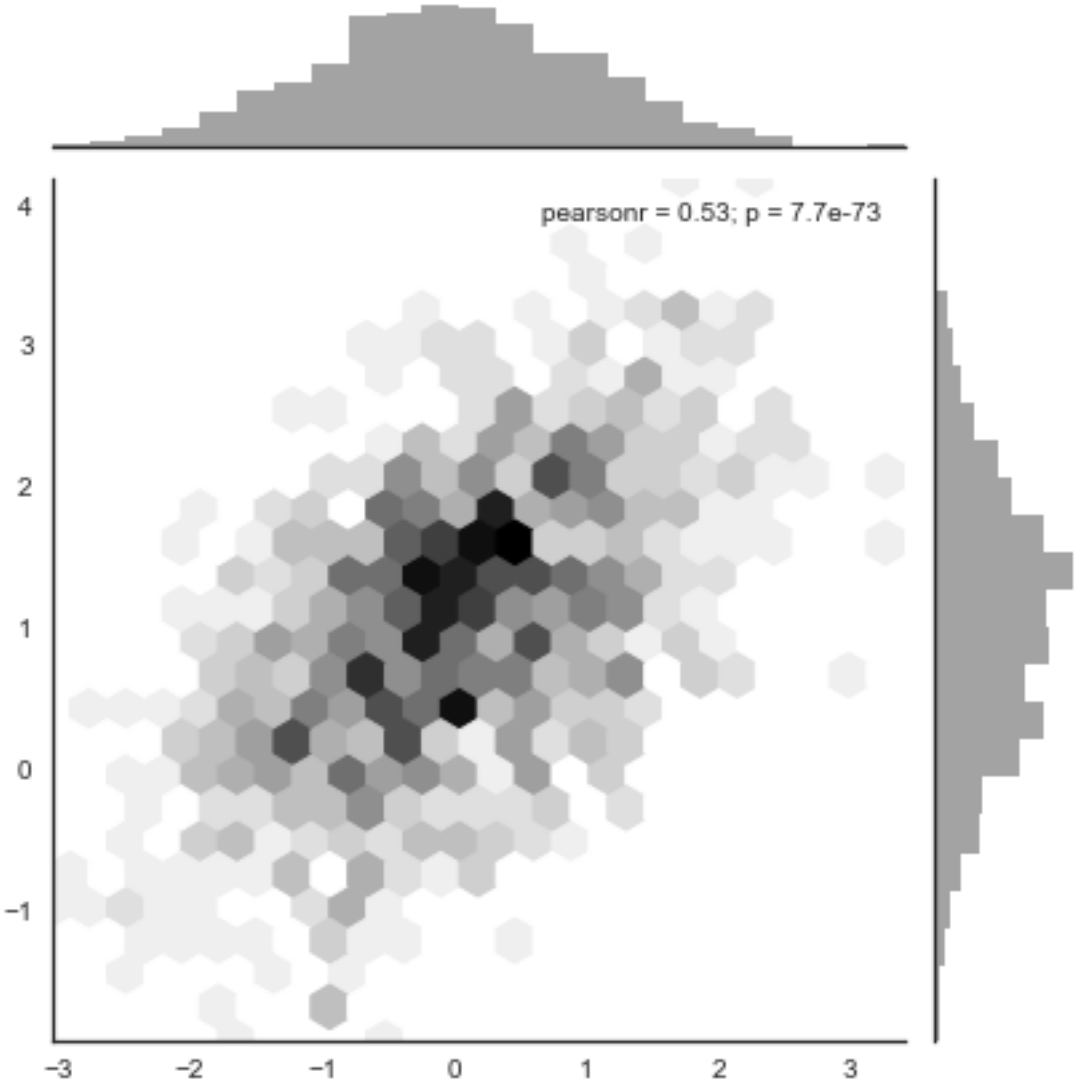
```
190 -0.288055  0.573457
191 -0.104574  1.322040
192  0.836075  2.449662
193 -1.207614  0.201229
194  1.035296  1.536901
195  0.831858  0.331377
196 -0.520707  0.116603
197  0.450051  1.462764
198 -0.262411  0.672233
199  0.792602  1.026313

[200 rows x 2 columns]
```

6.0.2 观测两个变量之间的分布关系最好用散点图

```
In [40]: sns.jointplot(x="x", y="y", data=df);
```

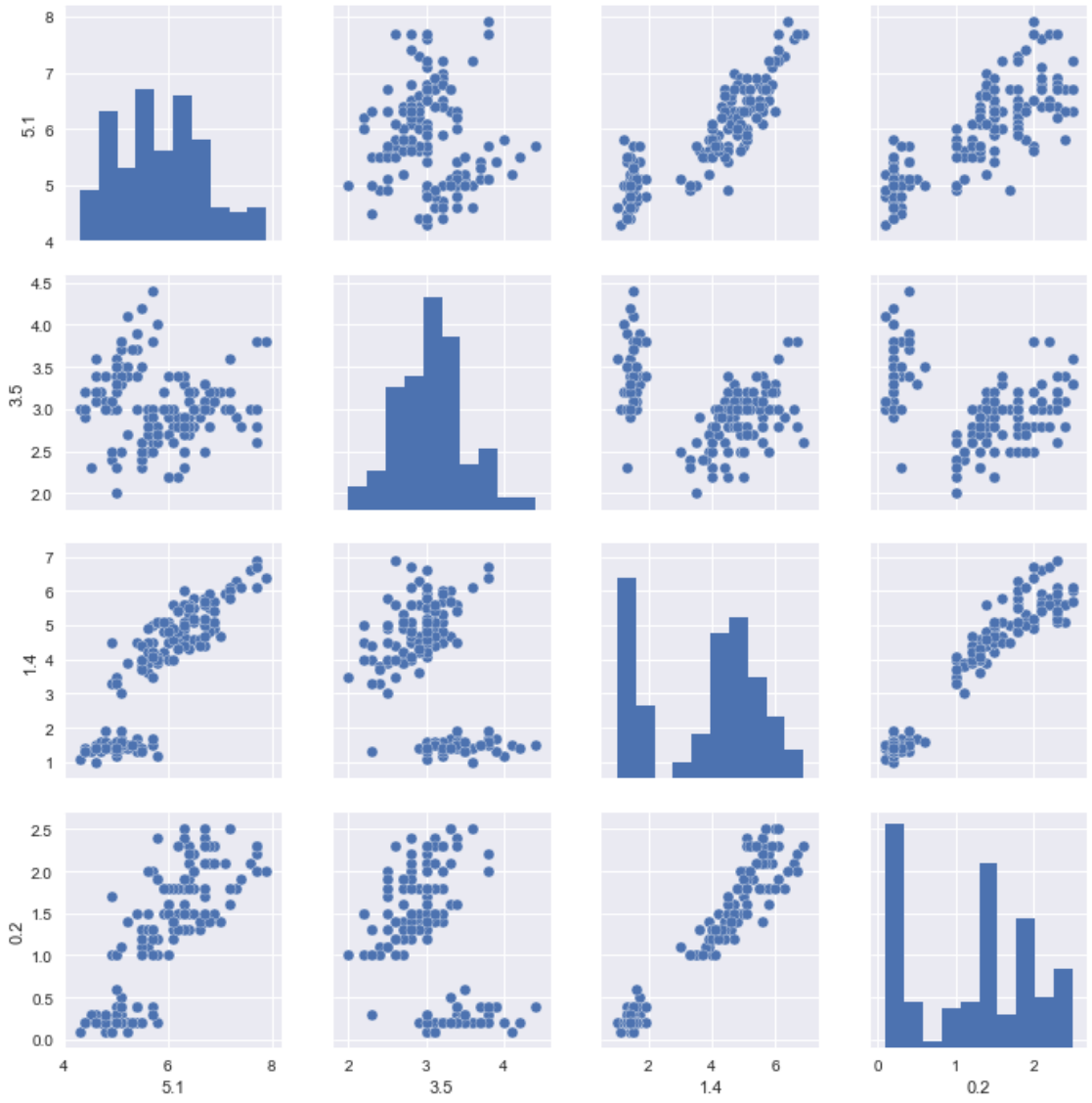



```
In [41]: x, y = np.random.multivariate_normal(mean, cov, 1000).T
         with sns.axes_style("white"):
             sns.jointplot(x=x, y=y, kind="hex", color="k")
```




```
In [42]: #iris = sns.load_dataset('iris')
         #sns.pairplot(iris)
         # 上面这两句执行不了所以用下面这两句
         iris = pd.read_csv('iris.csv')
         sns.pairplot(iris)
```

```
Out[42]: <seaborn.axisgrid.PairGrid at 0x1b6ca26e668>
```

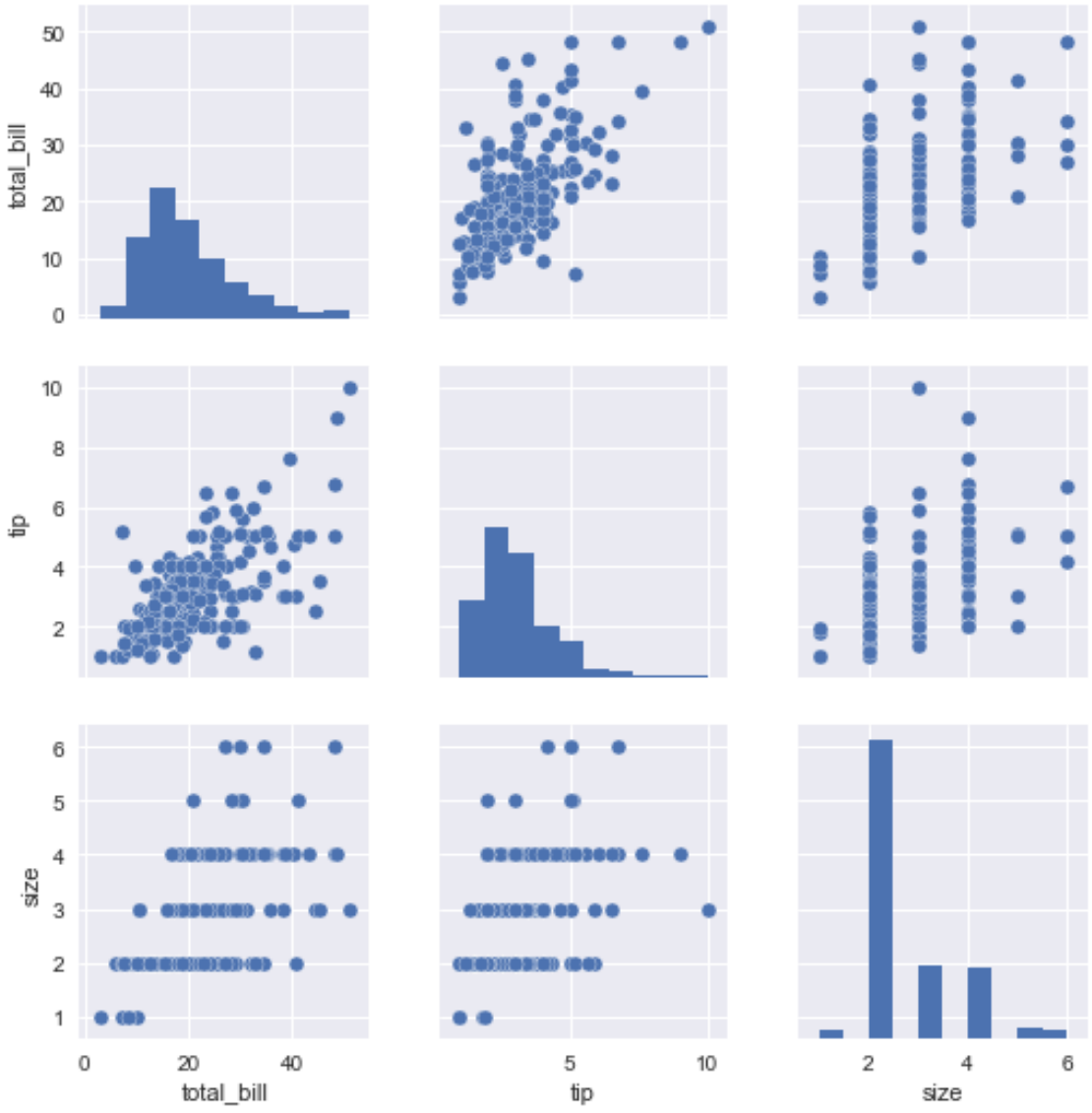



```
In [43]: %matplotlib inline
         import numpy as np
         import pandas as pd
         import matplotlib as mpl
         import matplotlib.pyplot as plt

         import seaborn as sns
         sns.set(color_codes=True)

         np.random.seed(sum(map(ord, "regression")))
```

```
#tips = sns.load_dataset("tips")

tips = pd.read_csv('tips.csv')
sns.pairplot(tips)

tips.head()
```

Out[43]:

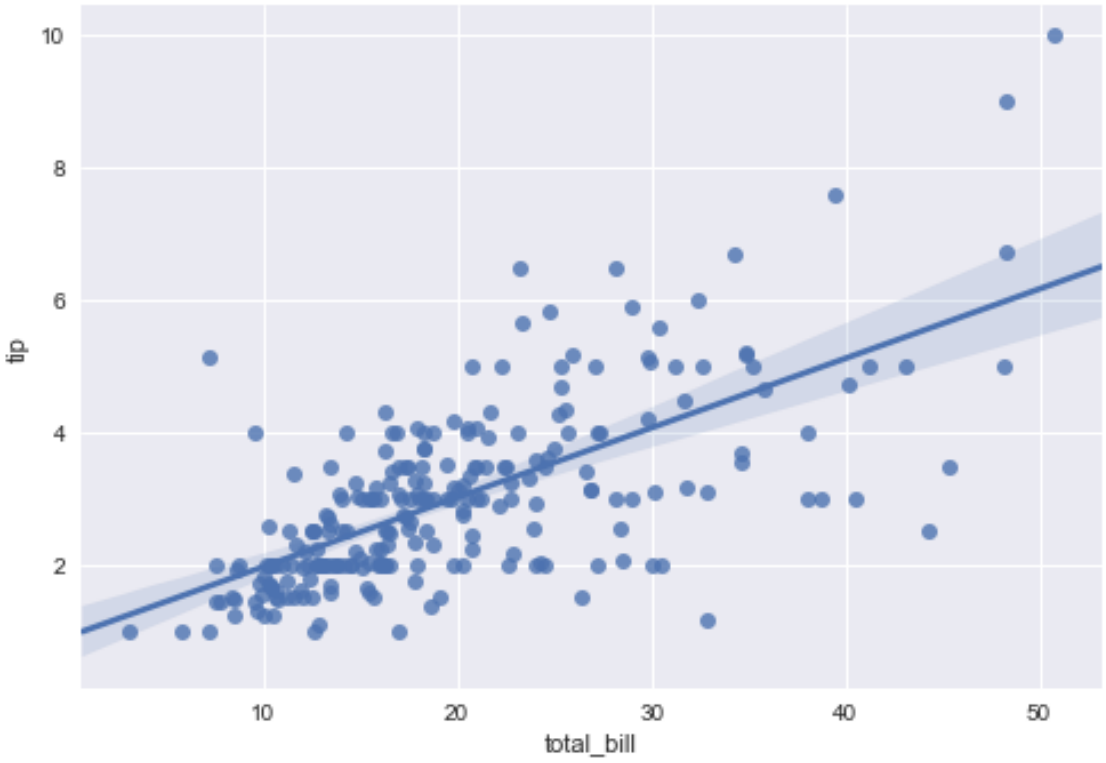
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4



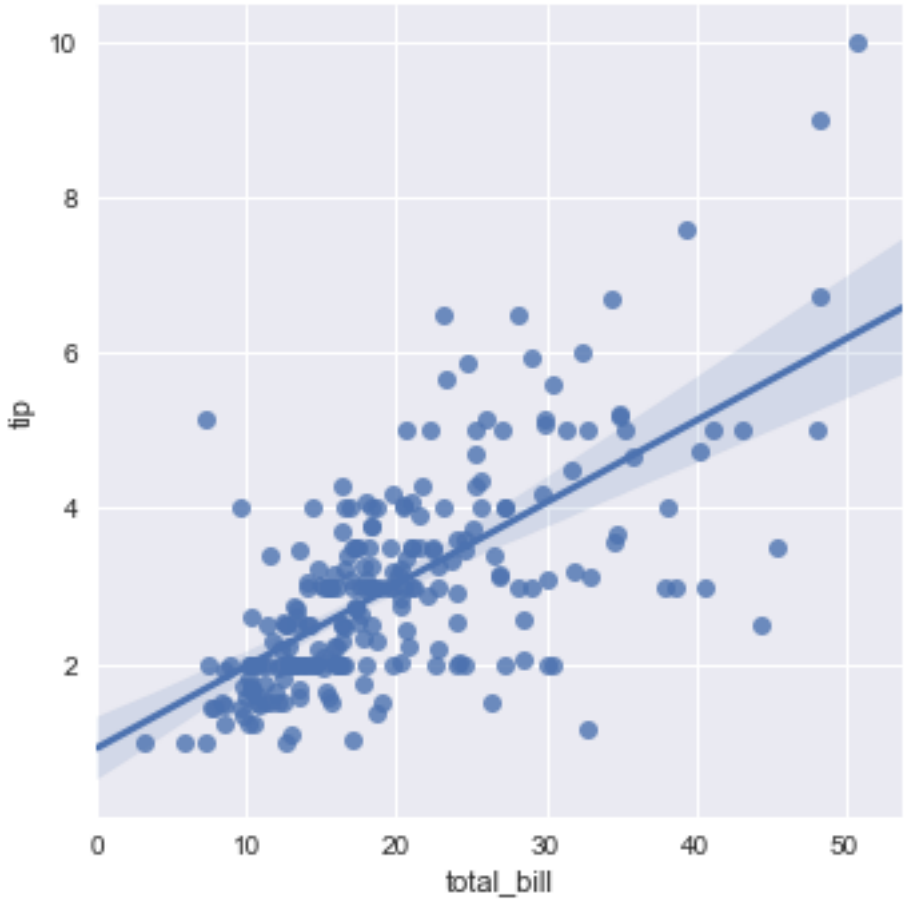
regplot() 和 lmplot() 都可以绘制回归关系, 推荐 regplot()

```
In [44]: sns.regplot(x="total_bill", y="tip", data=tips)
```

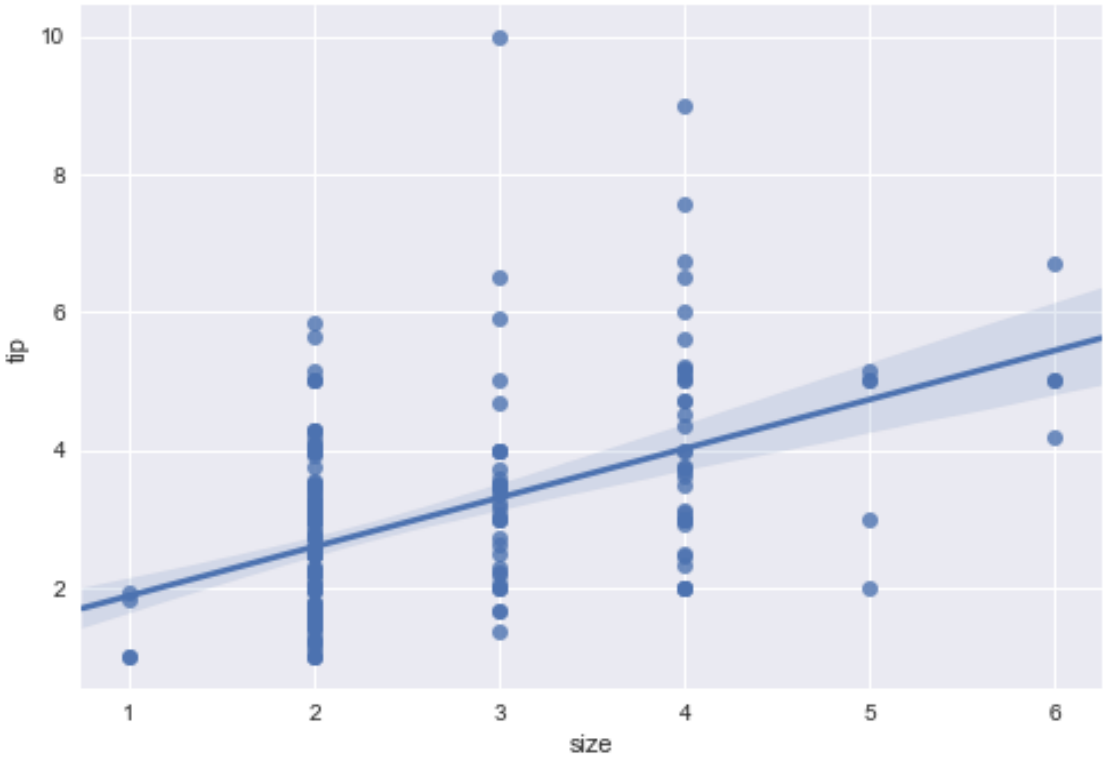
```
Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cb6d0550>
```



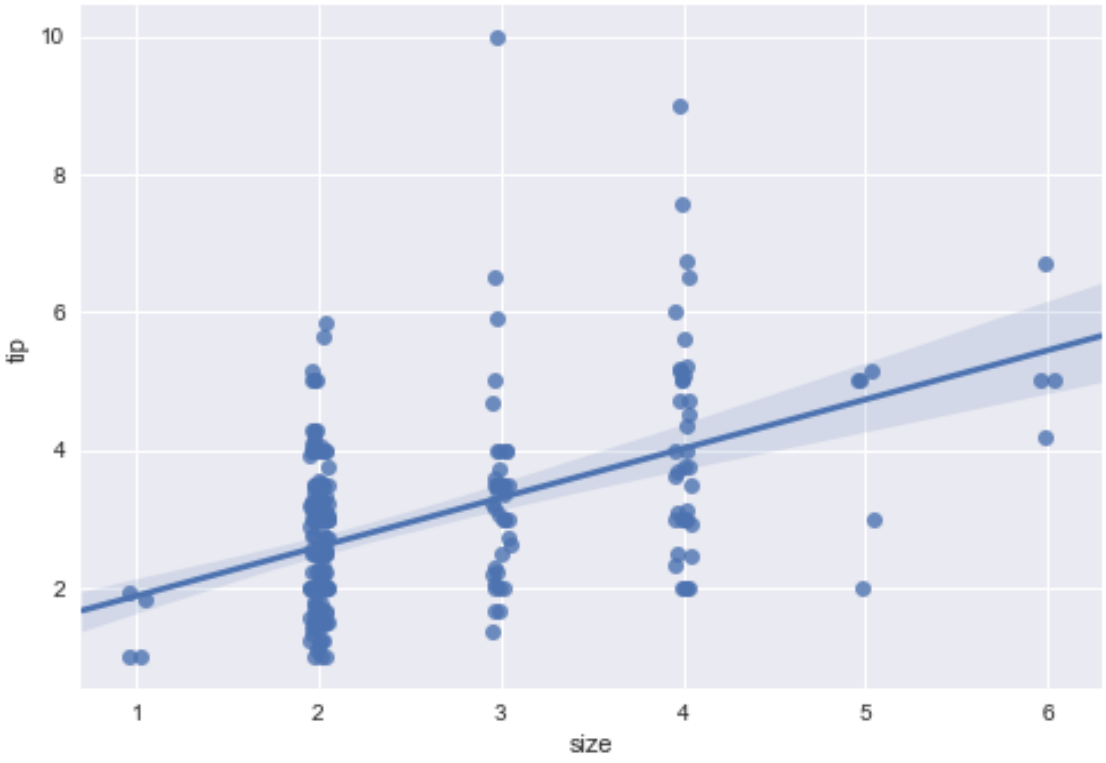

```
In [45]: sns.lmplot(x="total_bill", y="tip", data=tips);
```




```
In [46]: sns.regplot(data=tips,x="size",y="tip")

Out[46]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cc942748>
```

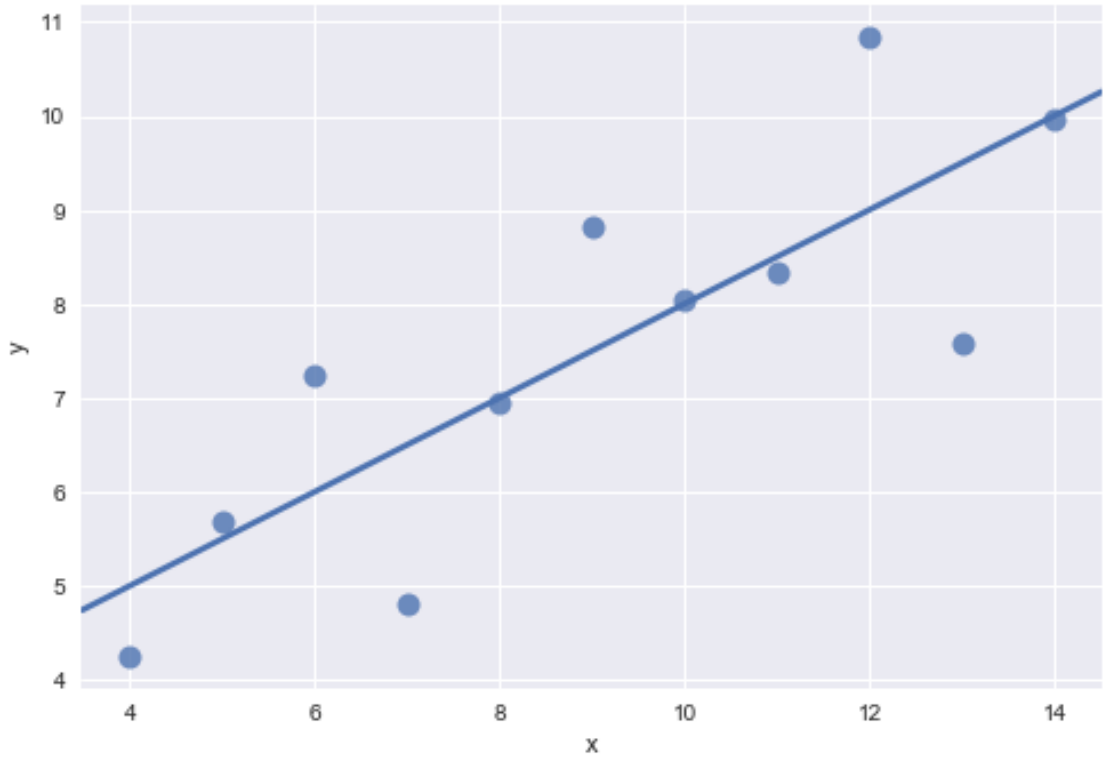



```
In [47]: sns.regplot(x="size", y="tip", data=tips, x_jitter=.05)

Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cc9e3550>
```



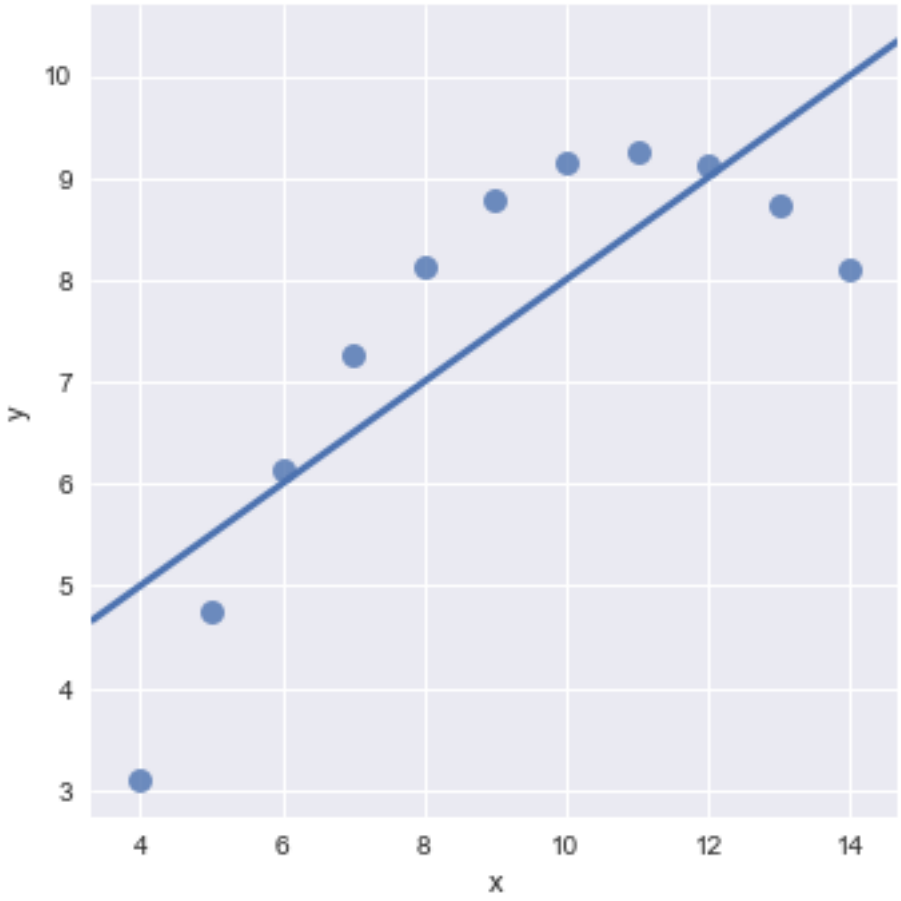

```
In [48]: #anscombe = sns.load_dataset("anscombe")
         anscombe = pd.read_csv('anscombe.csv')
         #sns.pairplot(anscombe)

         sns.regplot(x="x", y="y", data=anscombe.query("dataset == 'I'"),
                     ci=None, scatter_kws={"s": 100})
```

```
Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cca8dac8>
```

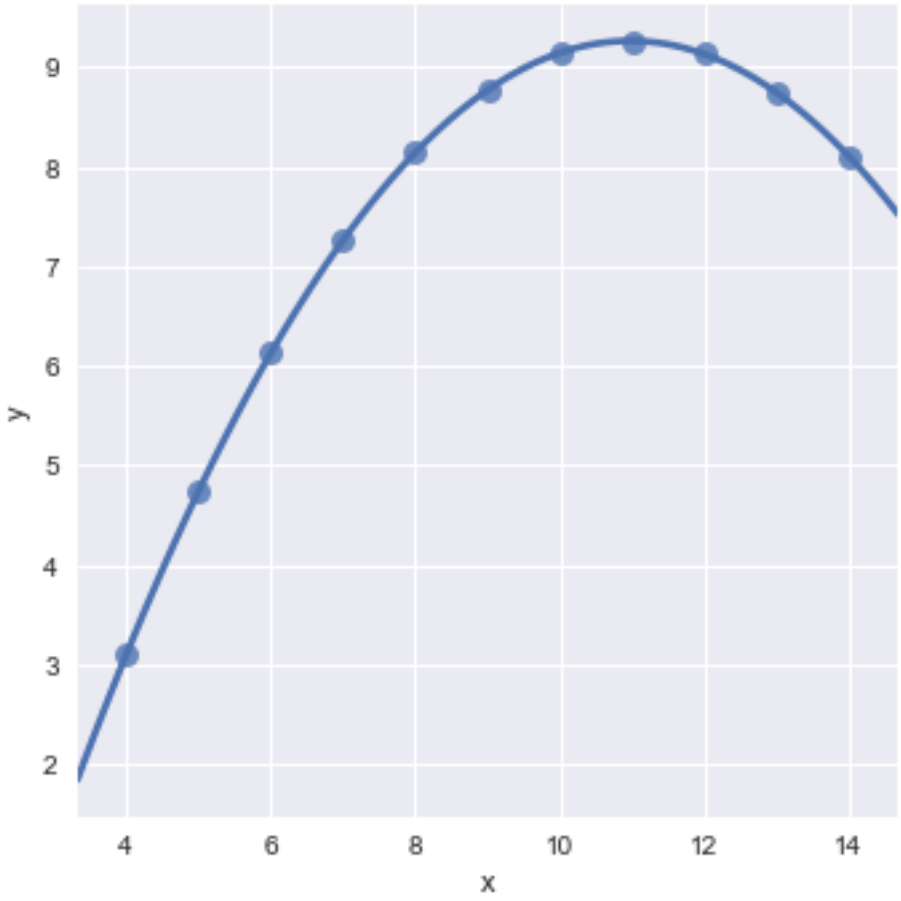



```
In [49]: sns.lmplot(x="x", y="y", data=anscombe.query("dataset == 'II'"),
                    ci=None, scatter_kws={"s": 80})
```

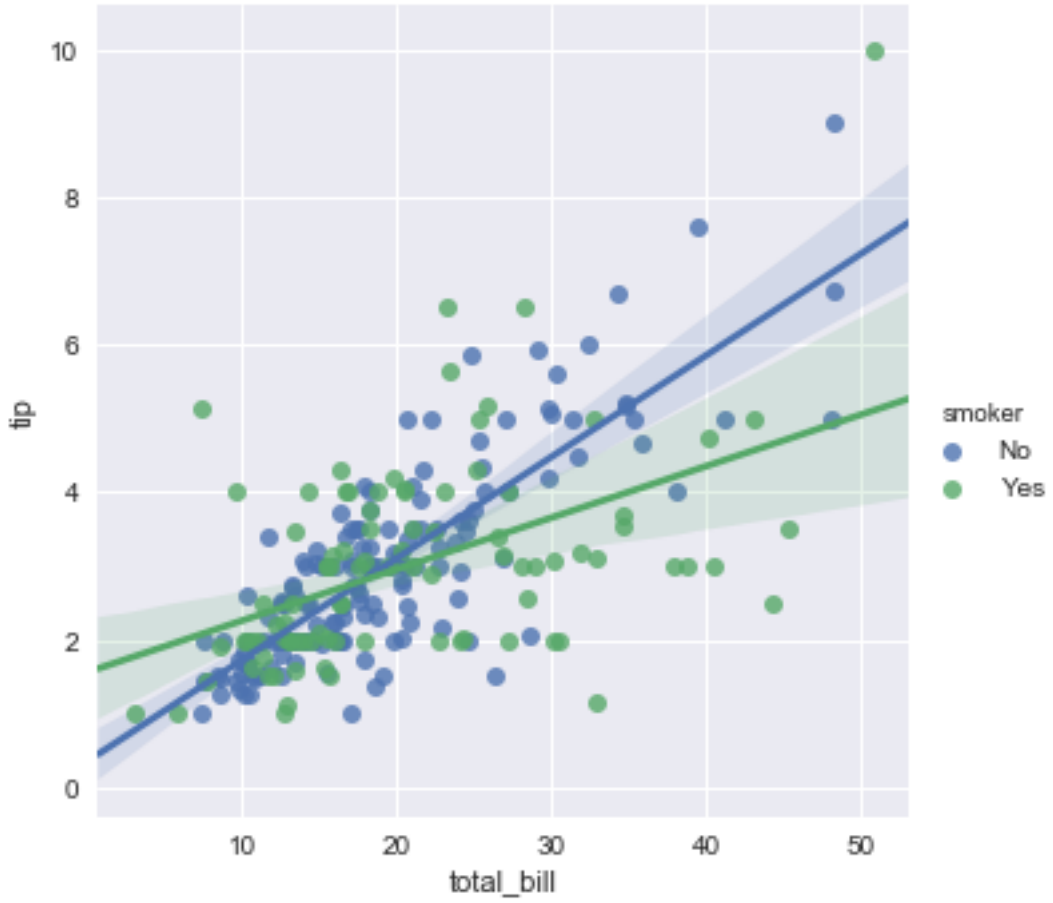
```
Out[49]: <seaborn.axisgrid.FacetGrid at 0x1b6ccb97518>
```



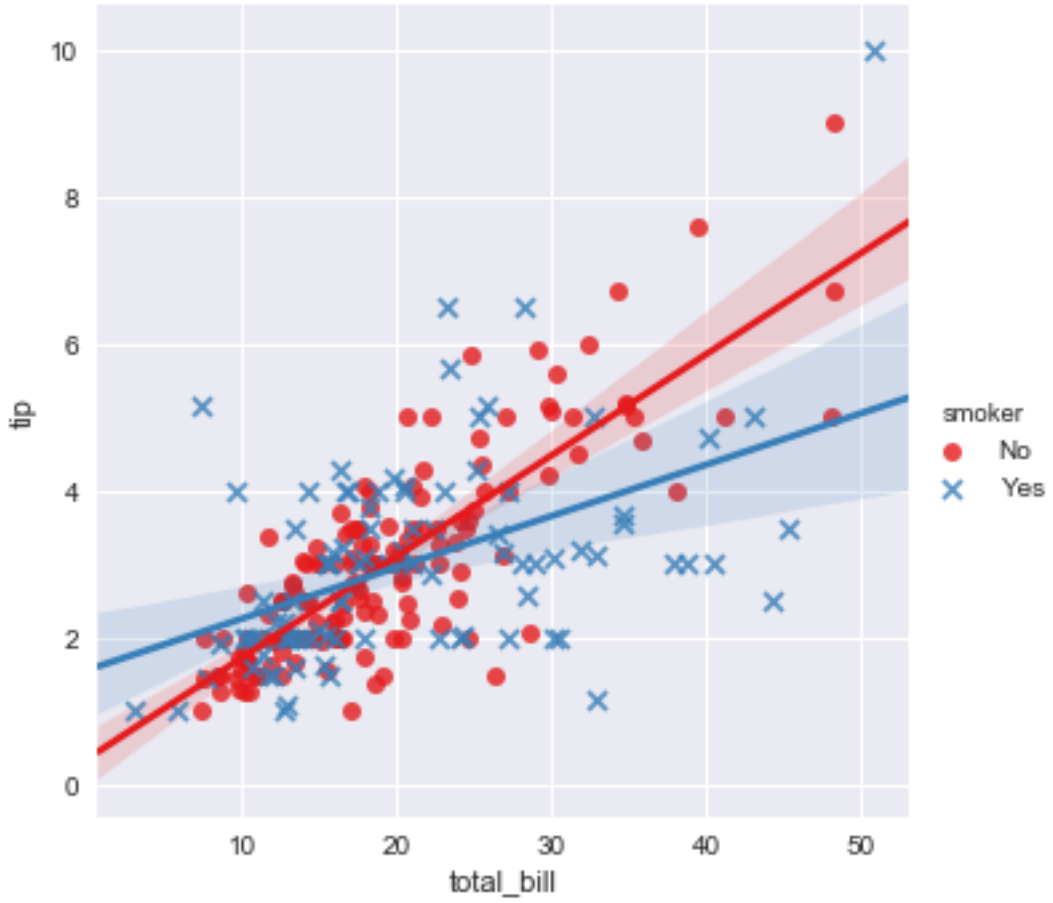

```
In [50]: sns.lmplot(x="x", y="y", data=anscombe.query("dataset == 'II'"),
                    order=2, ci=None, scatter_kws={"s": 80});
```



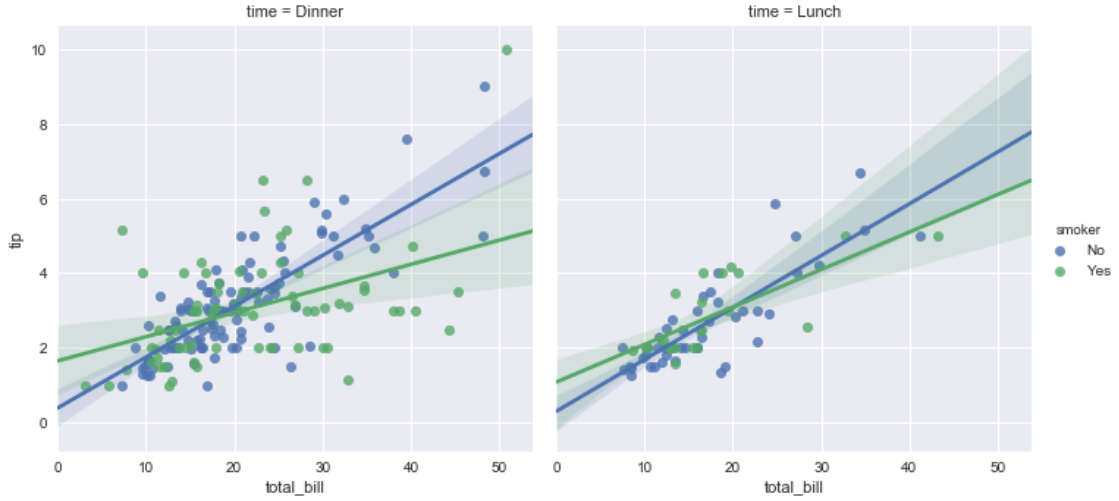

```
In [51]: sns.lmplot(x="total_bill", y="tip", hue="smoker", data=tips);
```



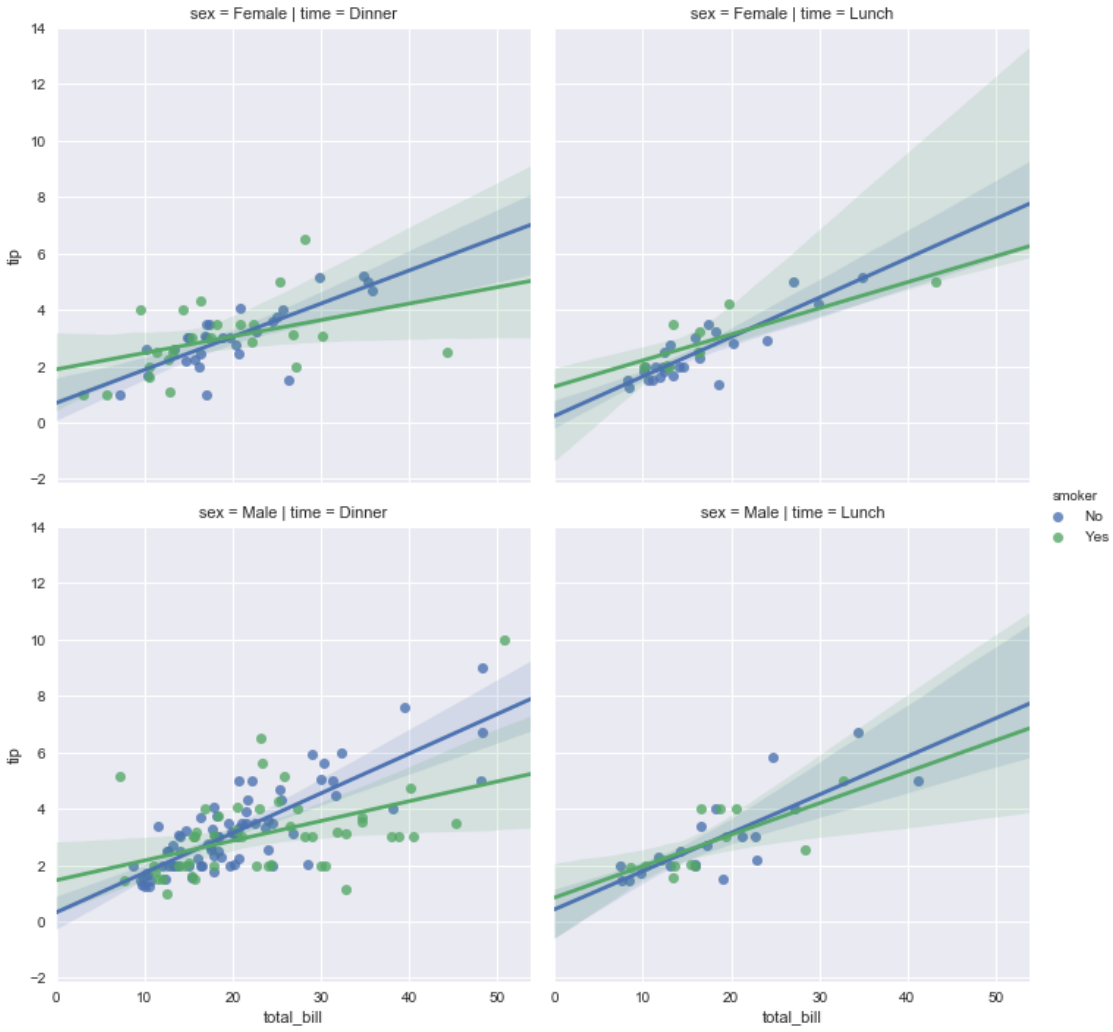

```
In [52]: sns.lmplot(x="total_bill", y="tip", hue="smoker", data=tips,
                    markers=["o", "x"], palette="Set1");
```



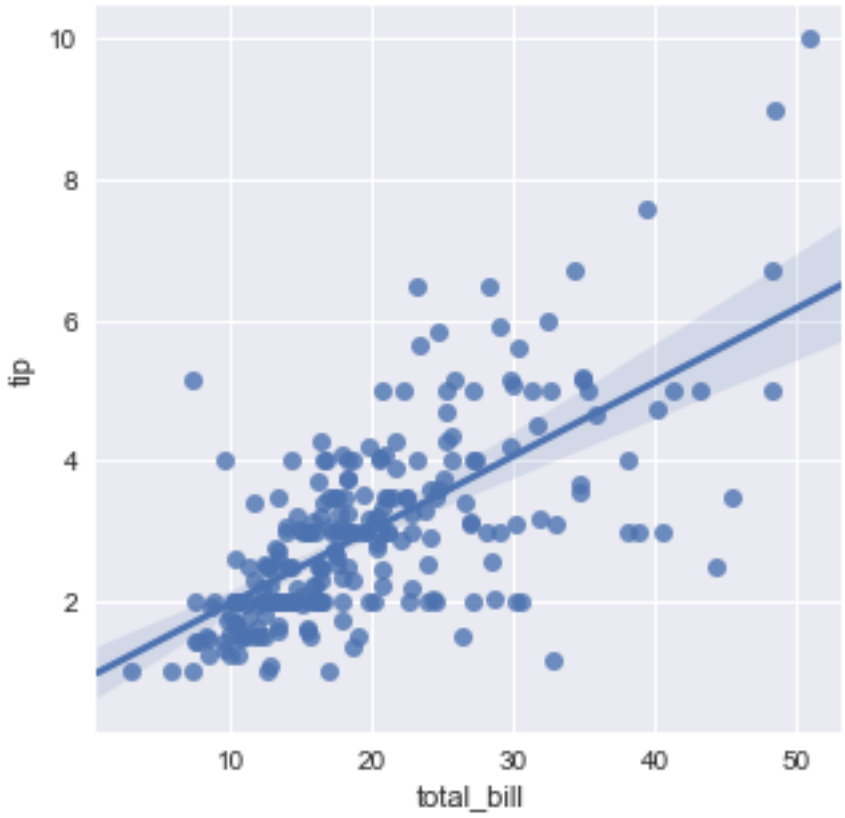

```
In [53]: sns.lmplot(x="total_bill", y="tip", hue="smoker", col="time", data=tips);
```



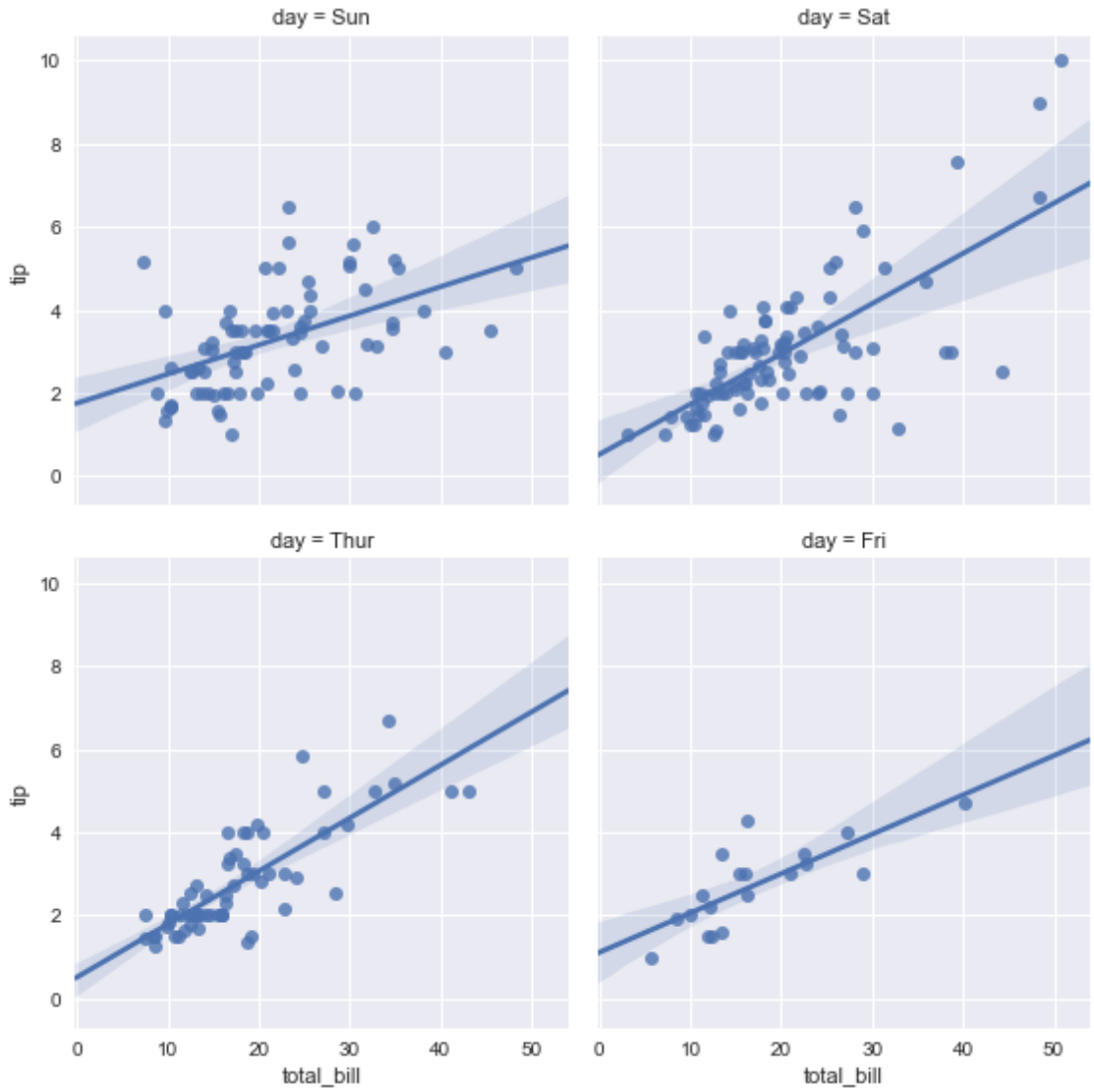
```
In [54]: sns.lmplot(x="total_bill", y="tip", hue="smoker",
                    col="time", row="sex", data=tips);
```



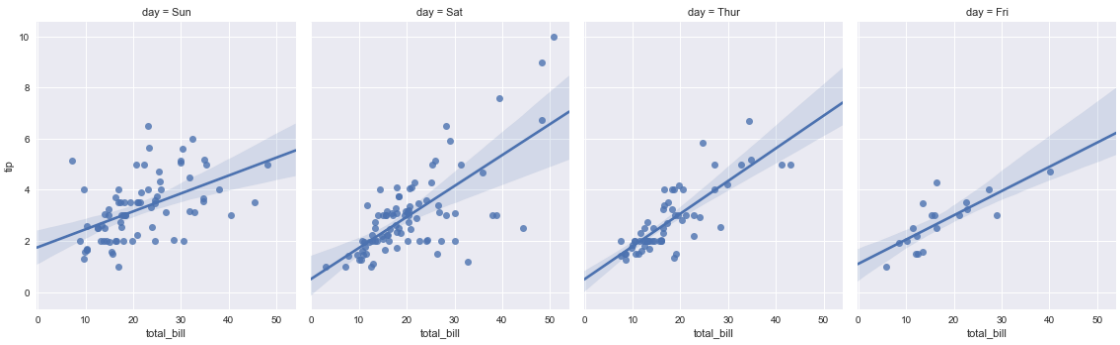

```
In [55]: f, ax = plt.subplots(figsize=(5, 5))
         sns.regplot(x="total_bill", y="tip", data=tips, ax=ax);
```




```
In [56]: sns.lmplot(x="total_bill", y="tip", col="day", data=tips,
                    col_wrap=2, size=4);
```

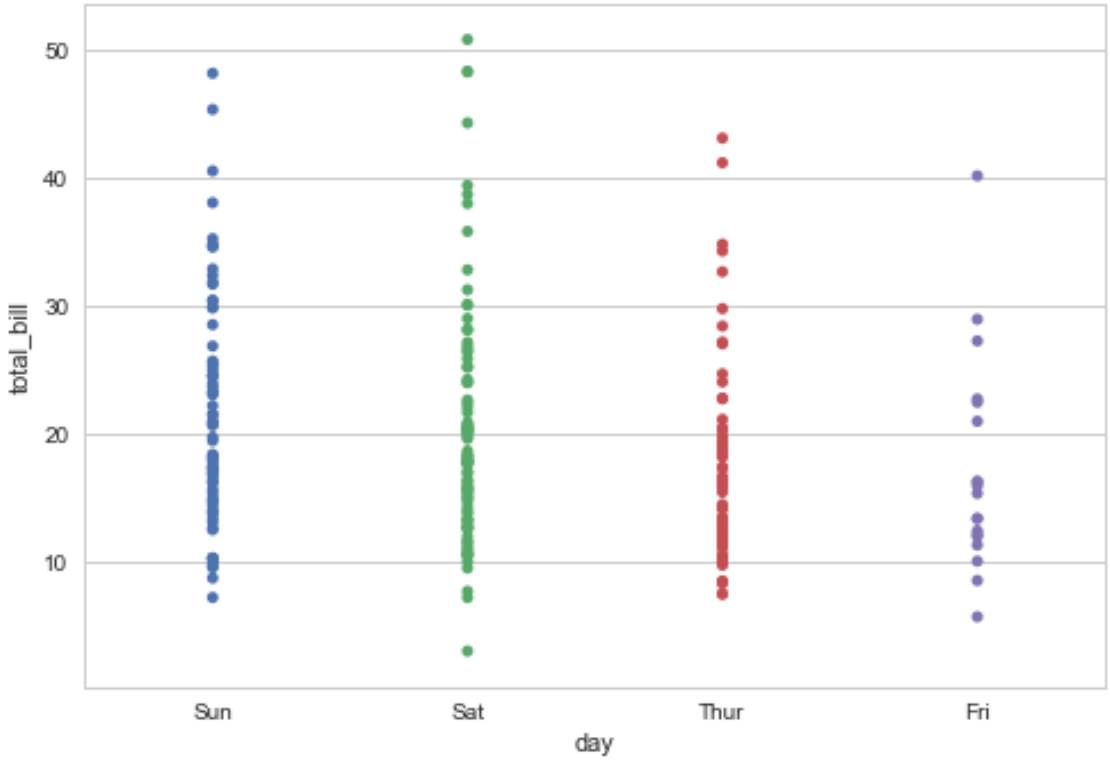



```
In [57]: sns.lmplot(x="total_bill", y="tip", col="day", data=tips,
                    aspect=.8);
```



In [58]:
```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid", color_codes=True)

np.random.seed(sum(map(ord, "categorical")))
#titanic = sns.load_dataset("titanic")
#tips = sns.load_dataset("tips")
#iris = sns.load_dataset("iris")

titanic = pd.read_csv('titanic.csv')
tips = pd.read_csv('tips.csv')
iris = pd.read_csv('iris.csv')
```

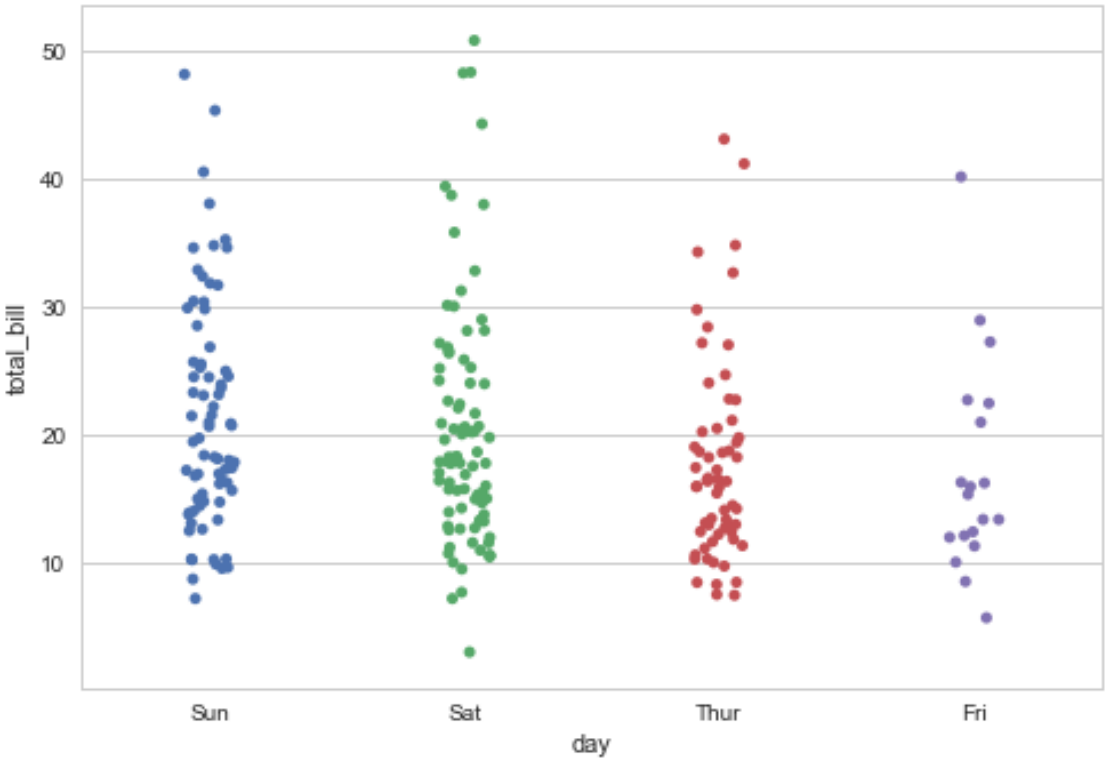
In [59]:
```
sns.stripplot(x="day", y="total_bill", data=tips);
```



重叠是很常见的现象，但是重叠影响我观察数据的量了

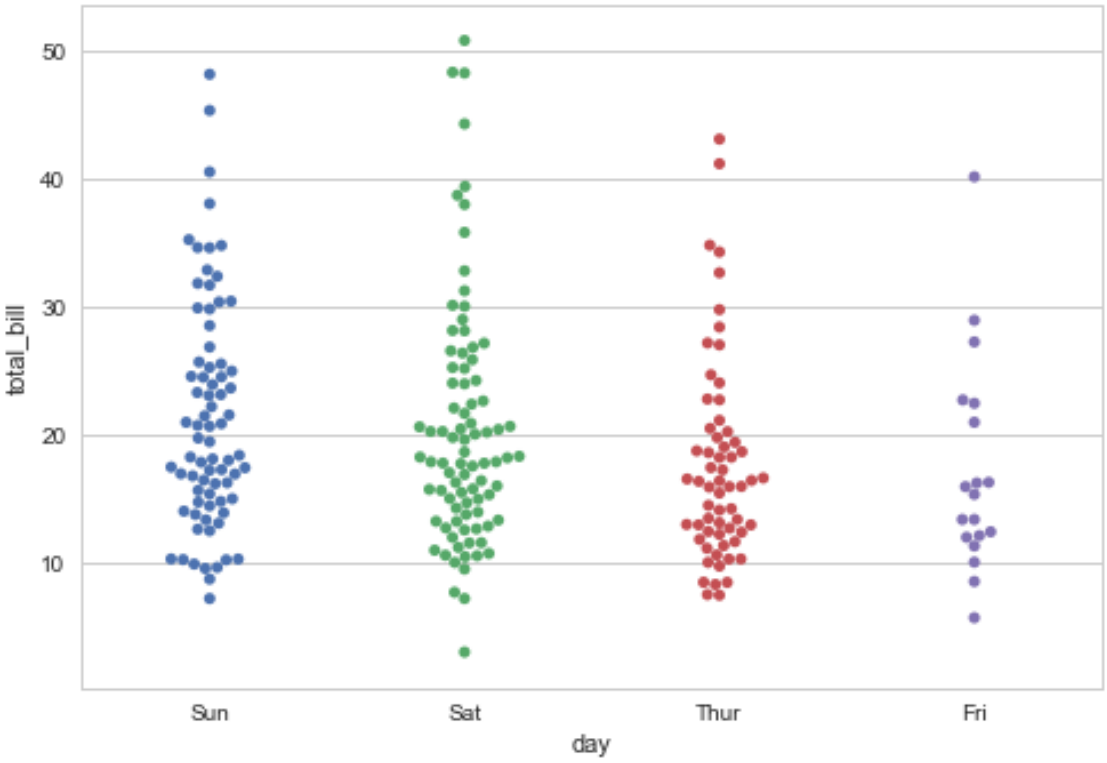
```
In [60]: sns.stripplot(x="day", y="total_bill", data=tips, jitter=True)
```

```
Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cb5dab70>
```

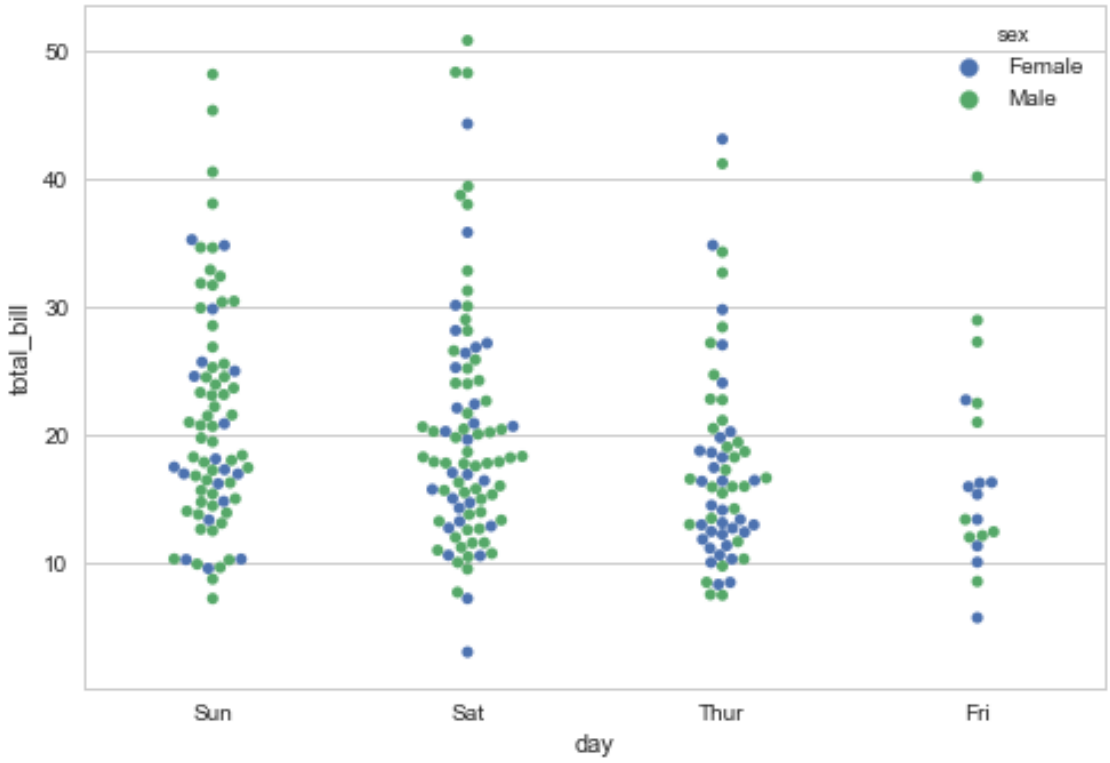



```
In [61]: sns.swarmplot(x="day", y="total_bill", data=tips)
```

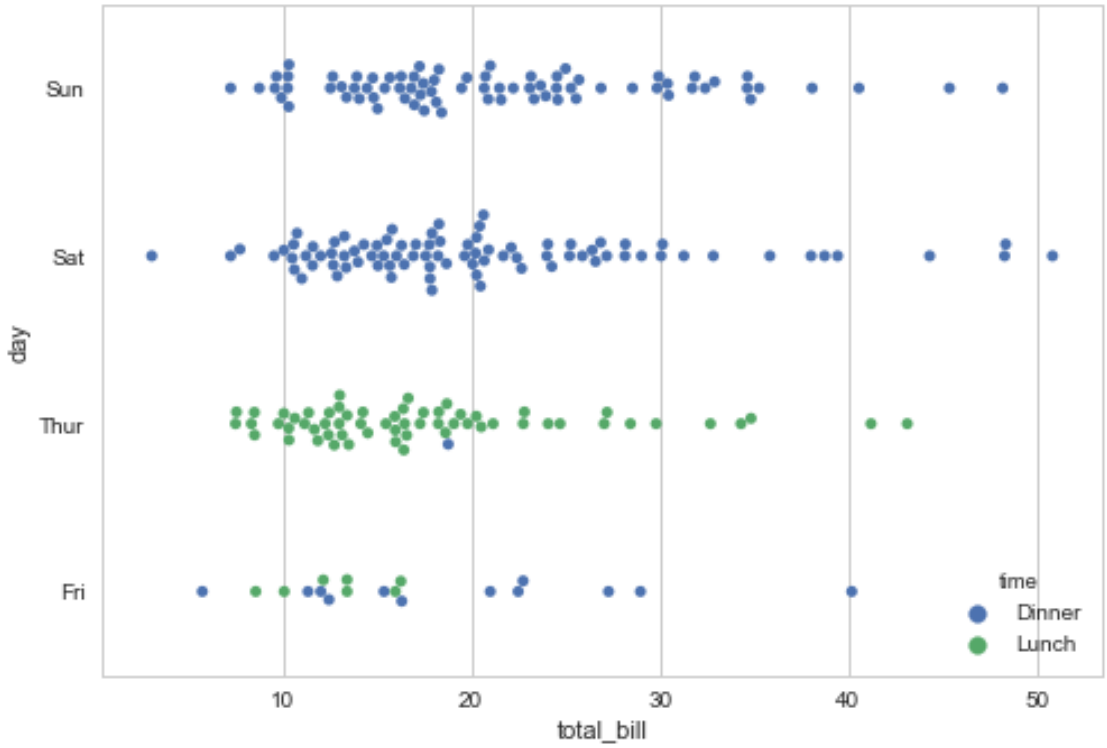
```
Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cb5ec240>
```




```
In [62]: sns.swarmplot(x="day", y="total_bill", hue="sex",data=tips)

Out[62]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6ca600c50>
```



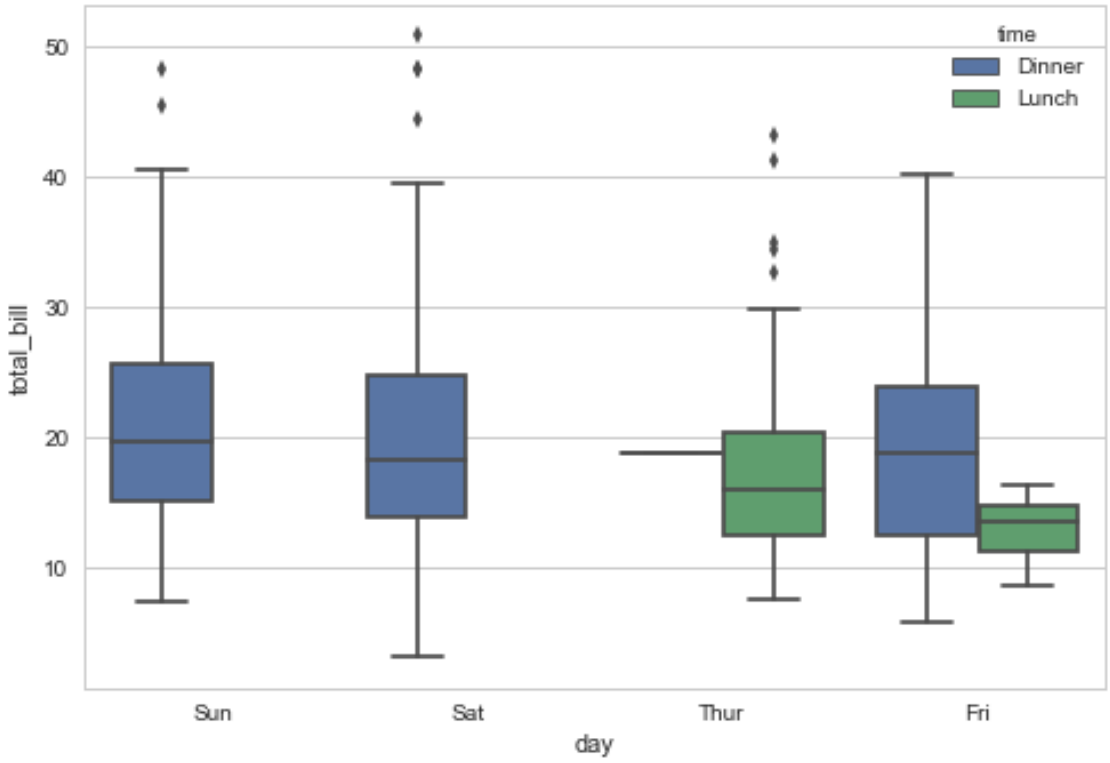

```
In [63]: sns.swarmplot(x="total_bill", y="day", hue="time", data=tips);
```



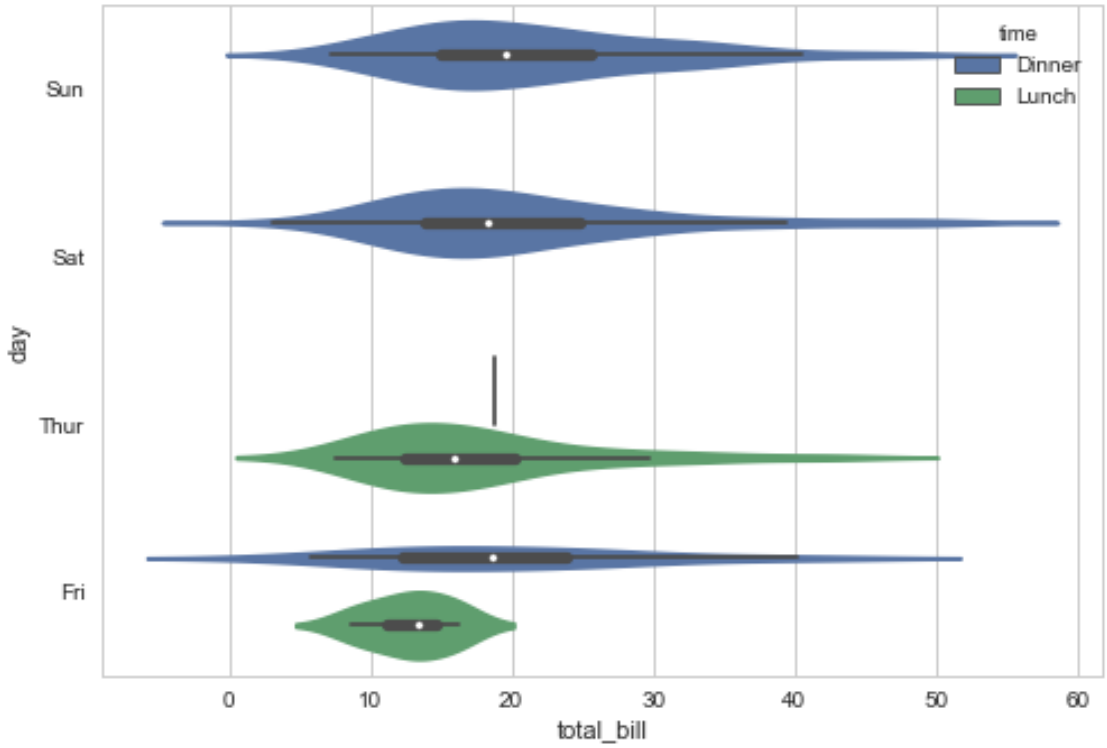
6.0.3 盒图

- IQR 即统计学概念四分位距，第一/四分位与第三/四分位之间的距离
- N = 1.5IQR 如果一个值 >Q3+N 或 < Q 1-N, 则为离群点

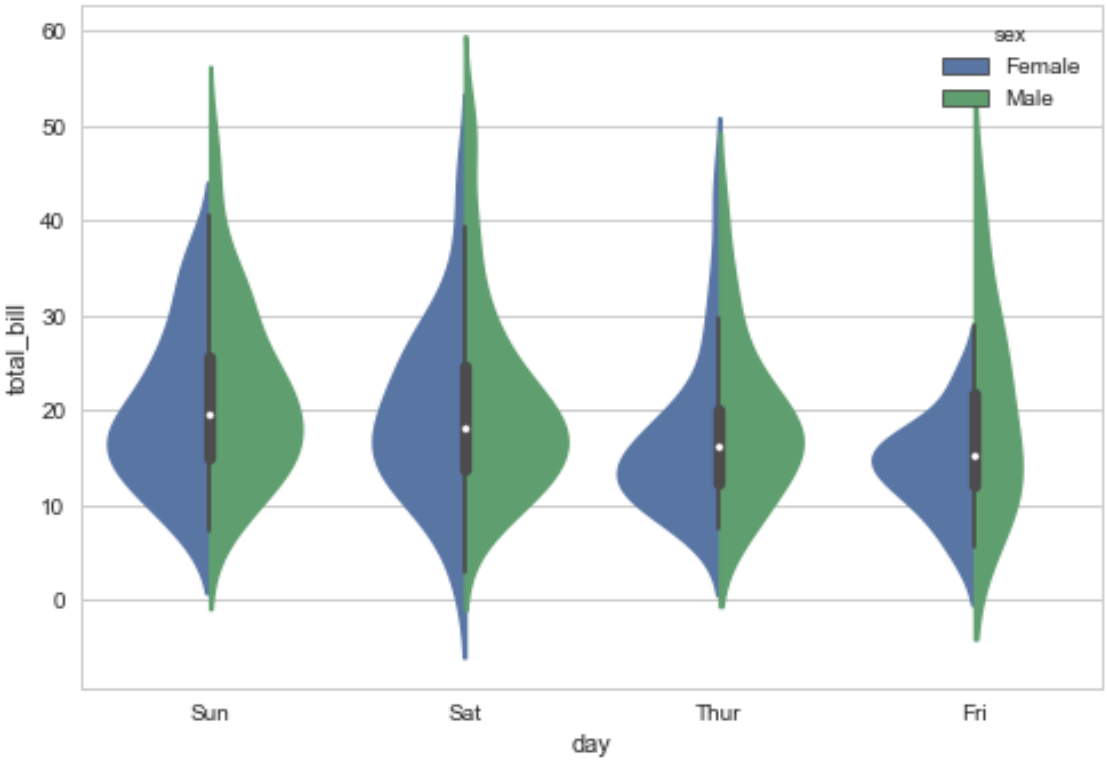
```
In [64]: sns.boxplot(x="day", y="total_bill", hue="time", data=tips);
```



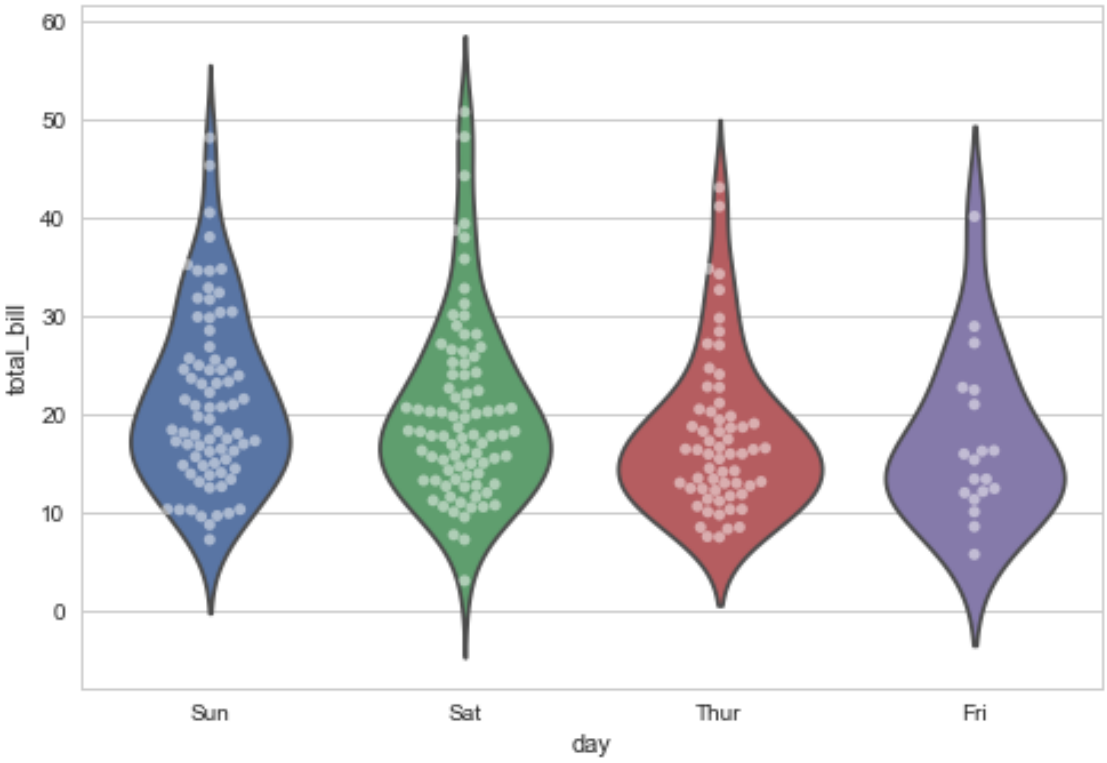

```
In [65]: sns.violinplot(x="total_bill", y="day", hue="time", data=tips);
```




```
In [66]: sns.violinplot(x="day", y="total_bill", hue="sex", data=tips, split=True);
```

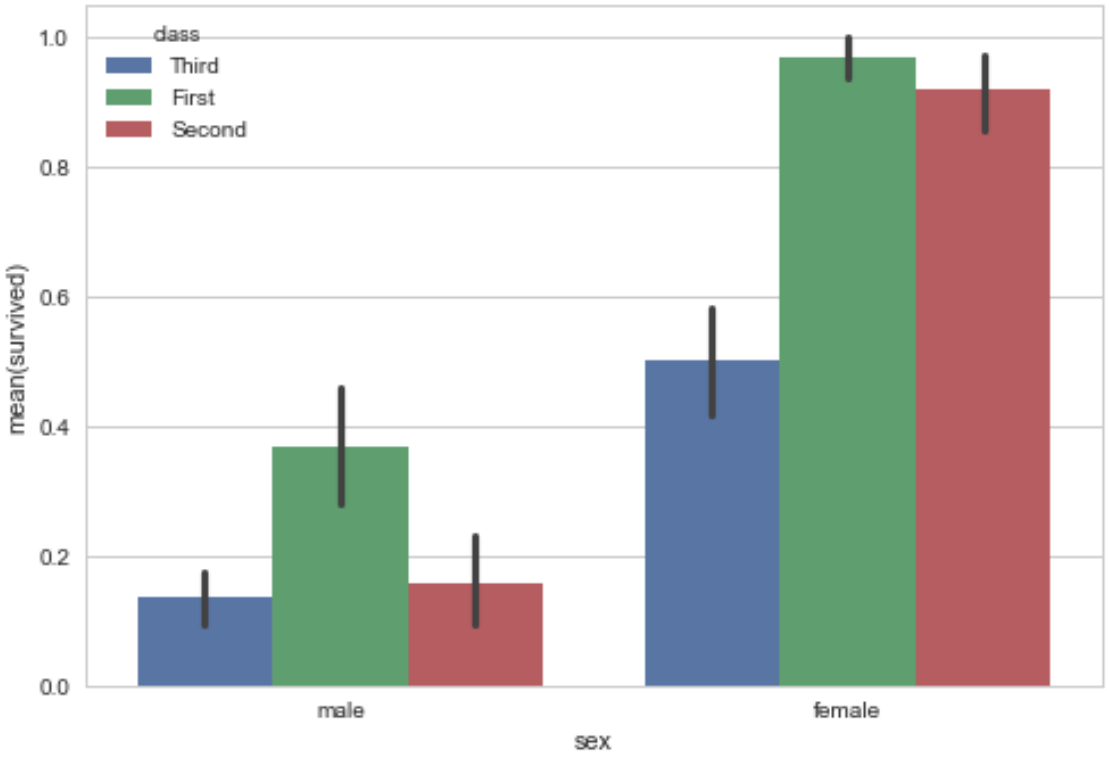



```
In [67]: sns.violinplot(x="day", y="total_bill", data=tips, inner=None)
         sns.swarmplot(x="day", y="total_bill", data=tips, color="w", alpha=.5)

Out[67]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6cd3fb898>
```



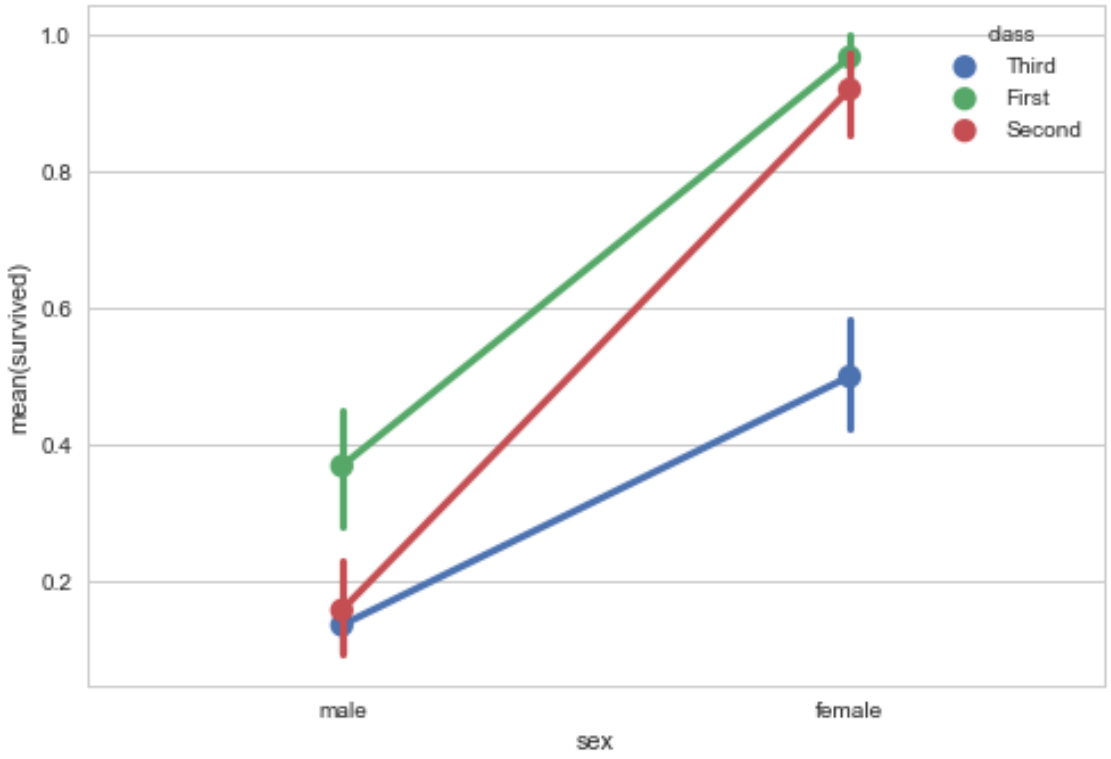
显示值的集中趋势可以用条形图

```
In [68]: sns.barplot(x="sex", y="survived", hue="class", data=titanic);
```

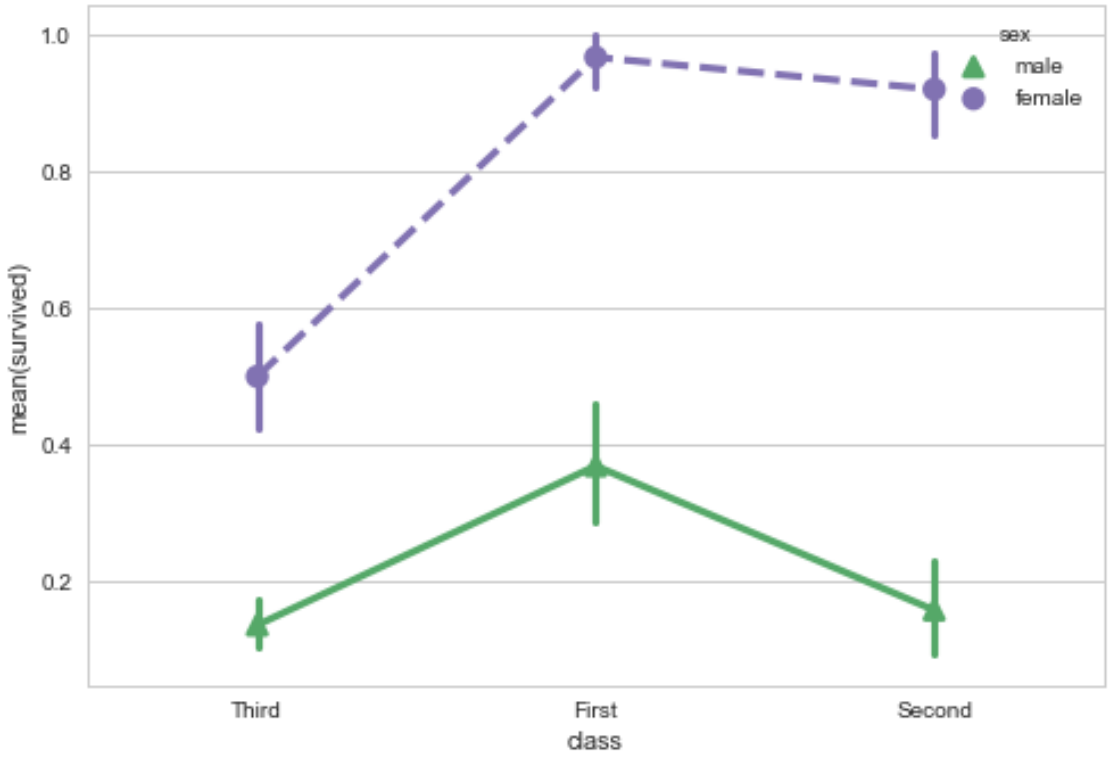


6.0.4 点图可以更好的描述变化差异

```
In [69]: sns.pointplot(x="sex", y="survived", hue="class", data=titanic);
```

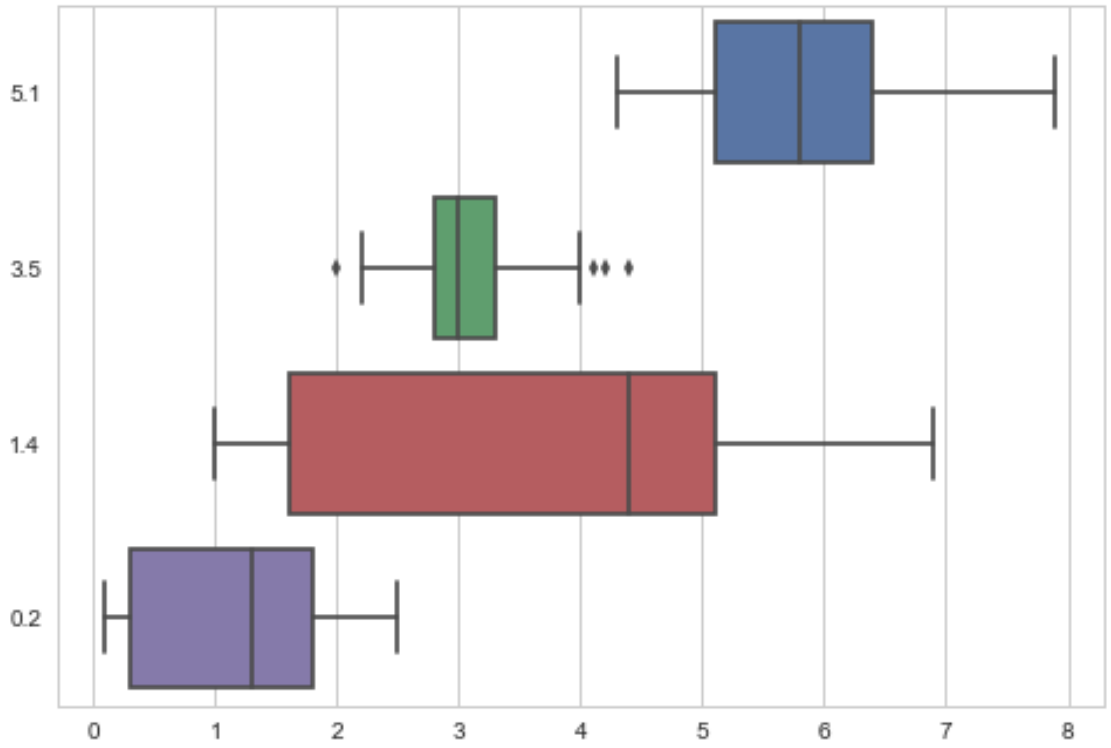



```
In [70]: sns.pointplot(x="class", y="survived", hue="sex", data=titanic,
                       palette={"male": "g", "female": "m"},
                       markers=["^", "o"], linestyles=["-", "--"]);
```



6.0.5 宽形数据

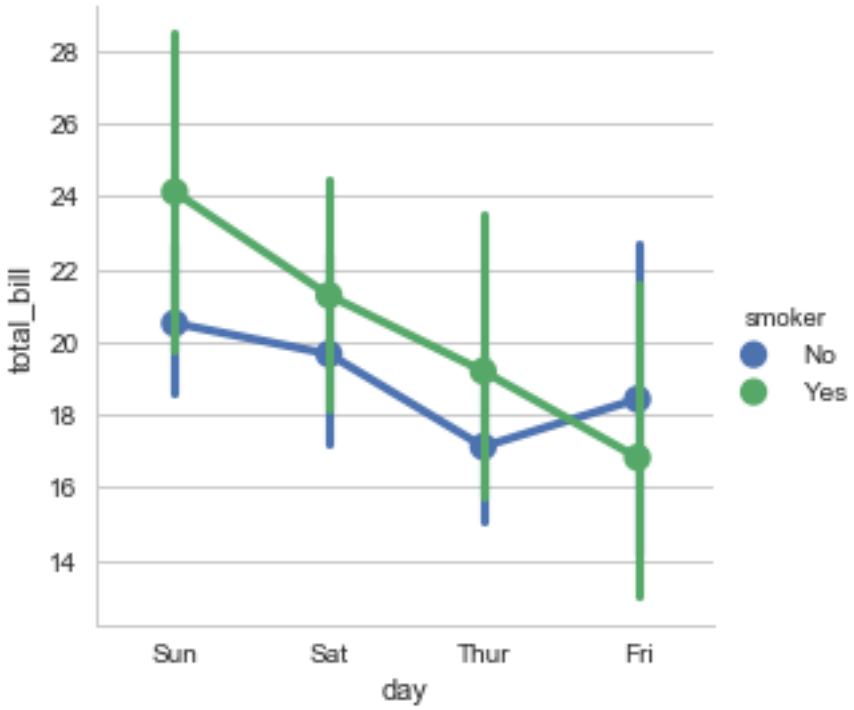
```
In [71]: sns.boxplot(data=iris,orient="h");
```



6.0.6 多层面板分类图

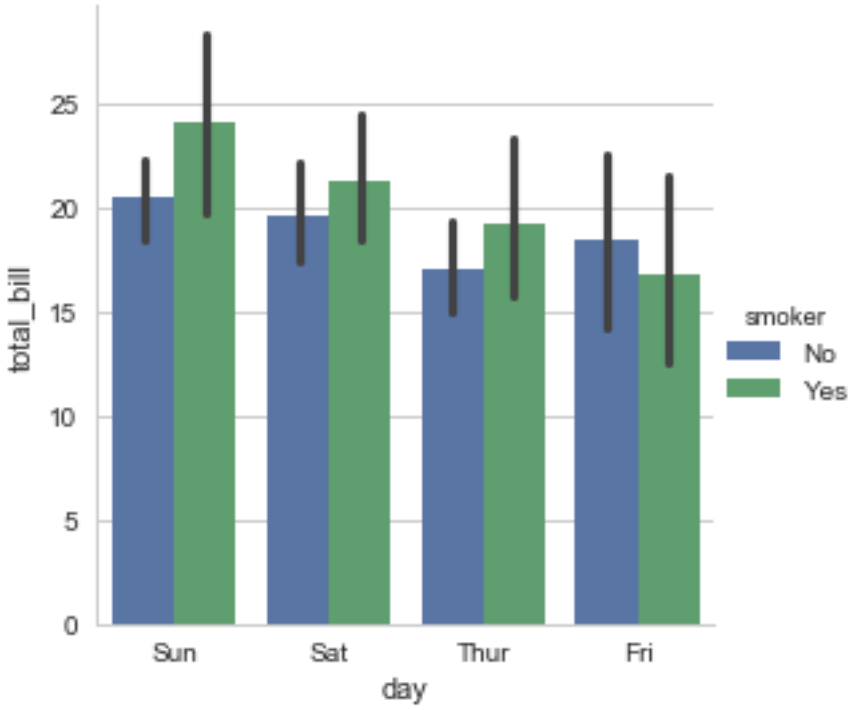
```
In [72]: sns.factorplot(x="day", y="total_bill", hue="smoker", data=tips)
```

```
Out[72]: <seaborn.axisgrid.FacetGrid at 0x1b6ccf105c0>
```



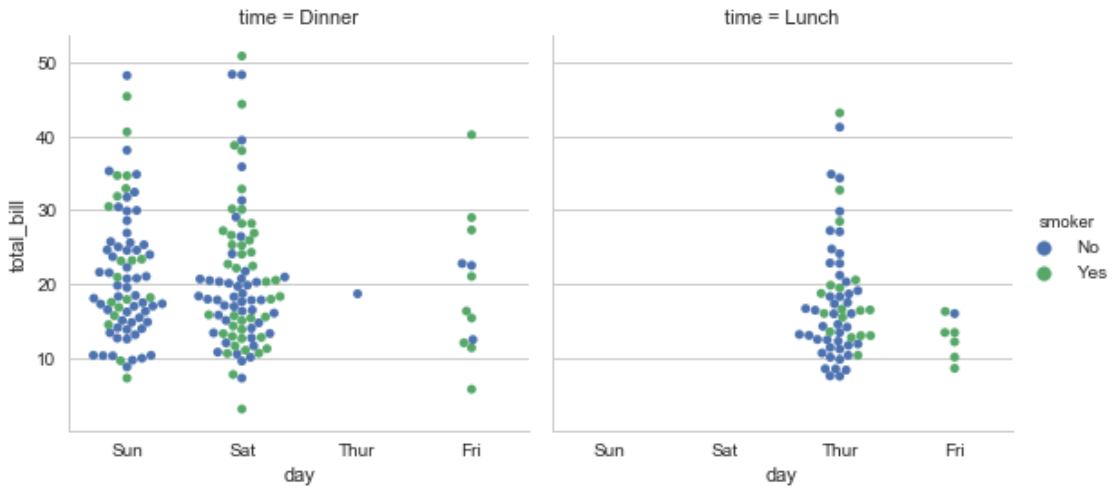

```
In [73]: sns.factorplot(x="day", y="total_bill", hue="smoker", data=tips, kind="bar")
```

```
Out[73]: <seaborn.axisgrid.FacetGrid at 0x1b6cb628048>
```



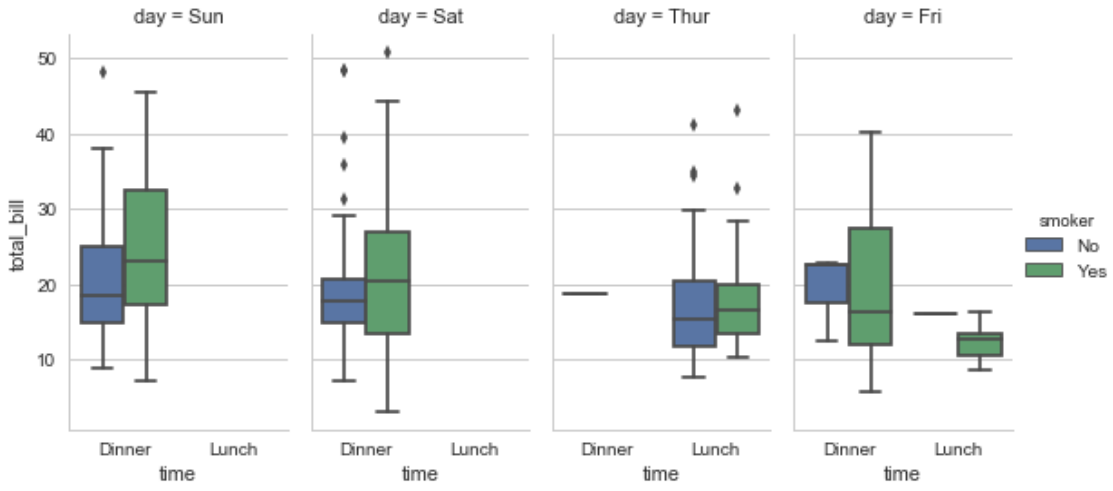
```
In [74]: sns.factorplot(x="day", y="total_bill", hue="smoker",
                        col="time", data=tips, kind="swarm")
```

```
Out[74]: <seaborn.axisgrid.FacetGrid at 0x1b6cb2fd3c8>
```




```
In [75]: sns.factorplot(x="time", y="total_bill", hue="smoker",
                        col="day", data=tips, kind="box", size=4, aspect=.5)
```

```
Out[75]: <seaborn.axisgrid.FacetGrid at 0x1b6cb3ea860>
```

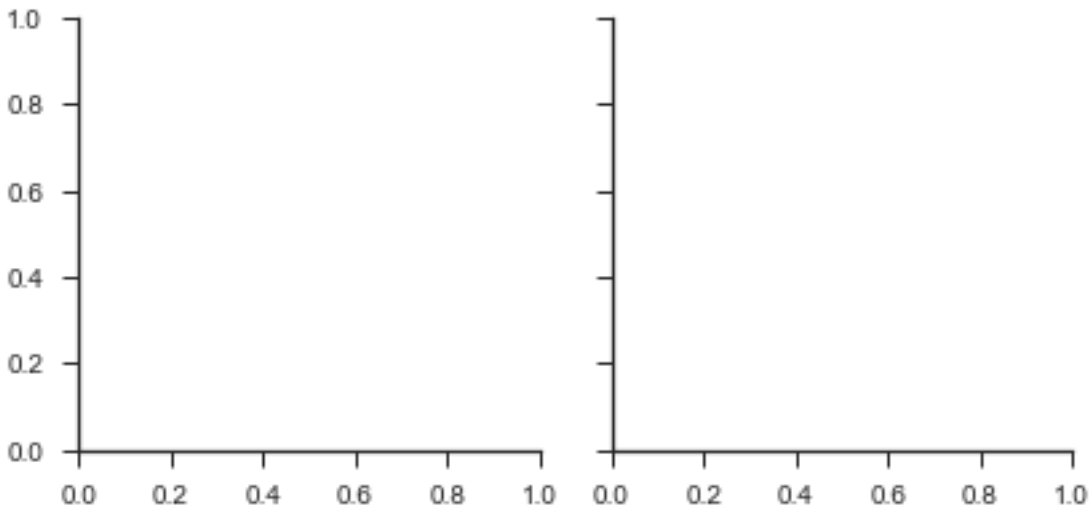


seaborn.factorplot(x=None, y=None, hue=None, data=None, row=None, col=None, col_wrap=None, estimator=, ci=95, n_boot=1000, units=None, order=None, hue_order=None, row_order=None, col_order=None, kind='point', size=4, aspect=1, orient=None, color=None, palette=None, legend=True, legend_out=True, sharex=True, sharey=True, margin_titles=False, facet_kws=None, **kwargs)

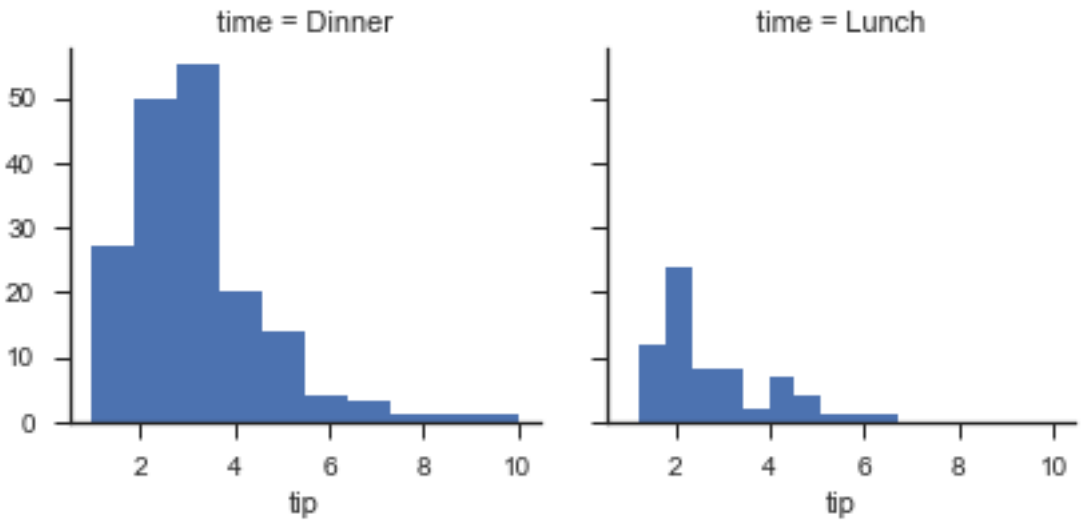
6.0.7 Parameters:

- x,y,hue 数据集变量变量名
- date 数据集数据集名
- row,col 更多分类变量进行平铺显示变量名
- col_wrap 每行的最高平铺数整数
- estimator 在每个分类中进行矢量到标量的映射矢量
- ci 置信区间浮点数或 None
- n_boot 计算置信区间时使用的引导迭代次数整数
- units 采样单元的标识符，用于执行多级引导和重复测量设计数据变量或向量数据
- order, hue_order 对应排序列表字符串列表
- row_order, col_order 对应排序列表字符串列表
- kind : 可选：point 默认, bar 柱形图, count 频次, box 箱体, violin 提琴, strip 散点，swarm 分散点 size 每个面的高度（英寸）标量 aspect 纵横比标量 orient 方向 "v"/"h" color 颜色 matplotlib 颜色 palette 调色板 seaborn 颜色色板或字典 legend hue 的信息面板 True/False legend_out 是否扩展图形，并将信息框绘制在中心右边 True/False share{x,y} 共享轴线 True/False

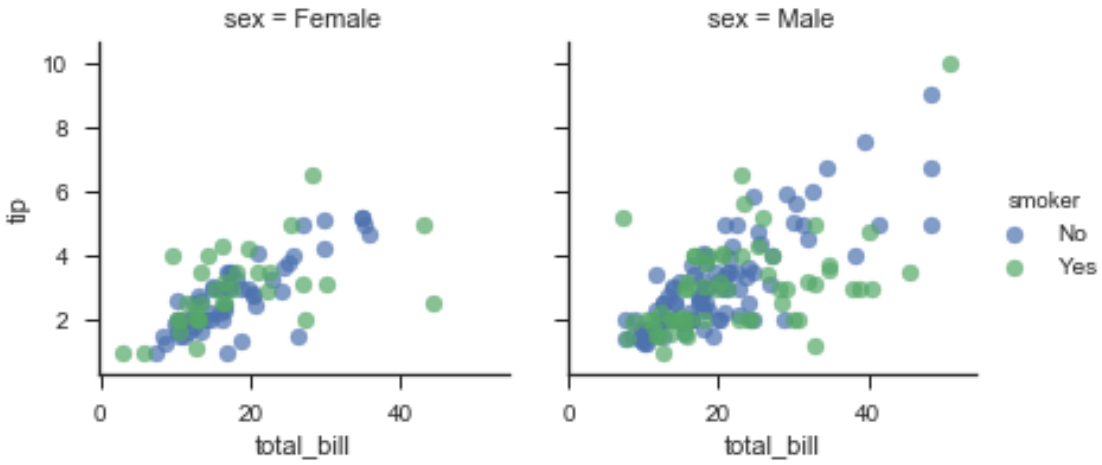
```
In [76]: %matplotlib inline
         import numpy as np
         import pandas as pd
         import seaborn as sns
         from scipy import stats
         import matplotlib as mpl
         import matplotlib.pyplot as plt

         sns.set(style="ticks")
         np.random.seed(sum(map(ord, "axis_grids")))

In [77]: g = sns.FacetGrid(tips, col="time")
```



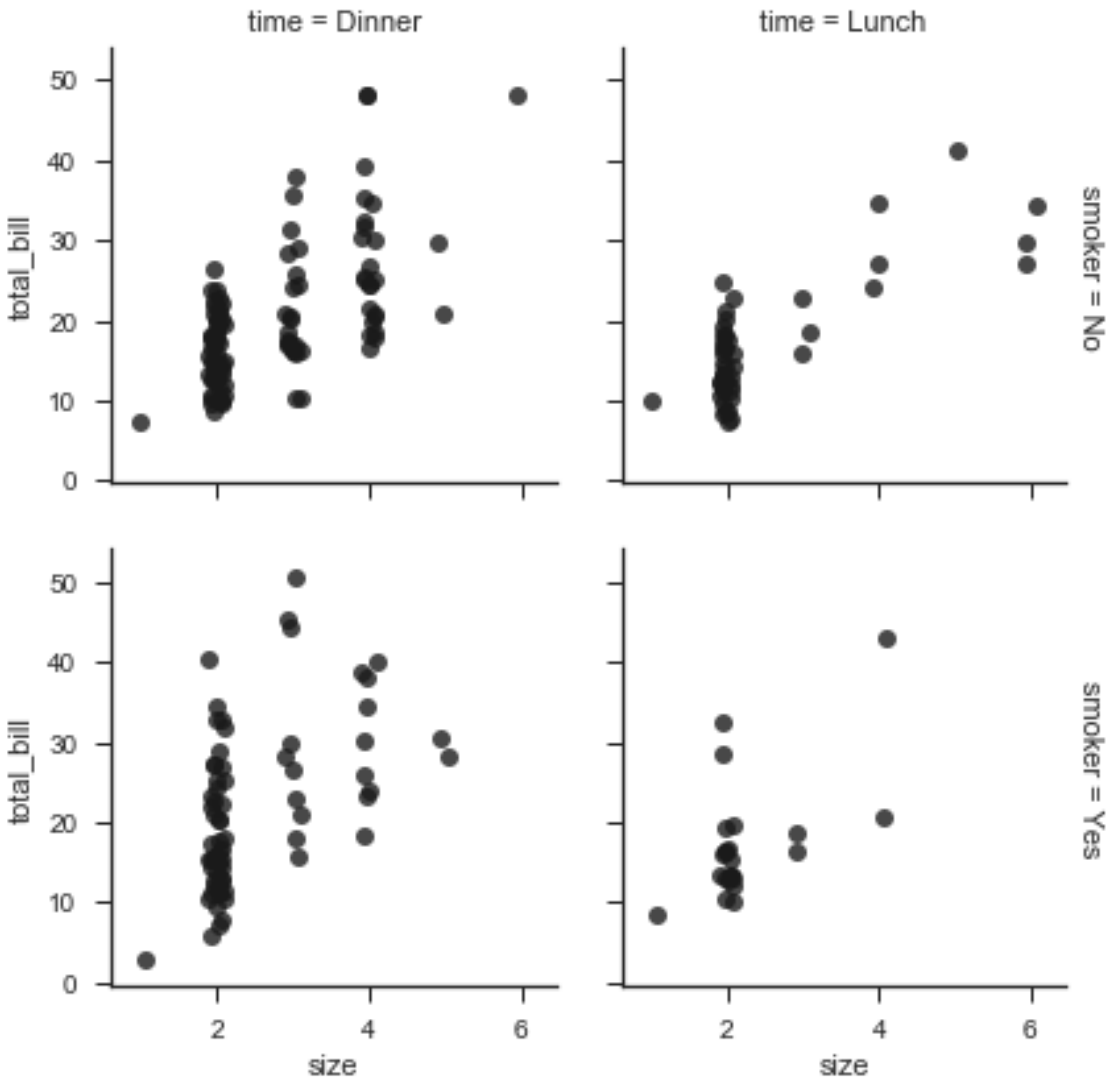

```
In [78]: g = sns.FacetGrid(tips, col="time")
         g.map(plt.hist, "tip");
```



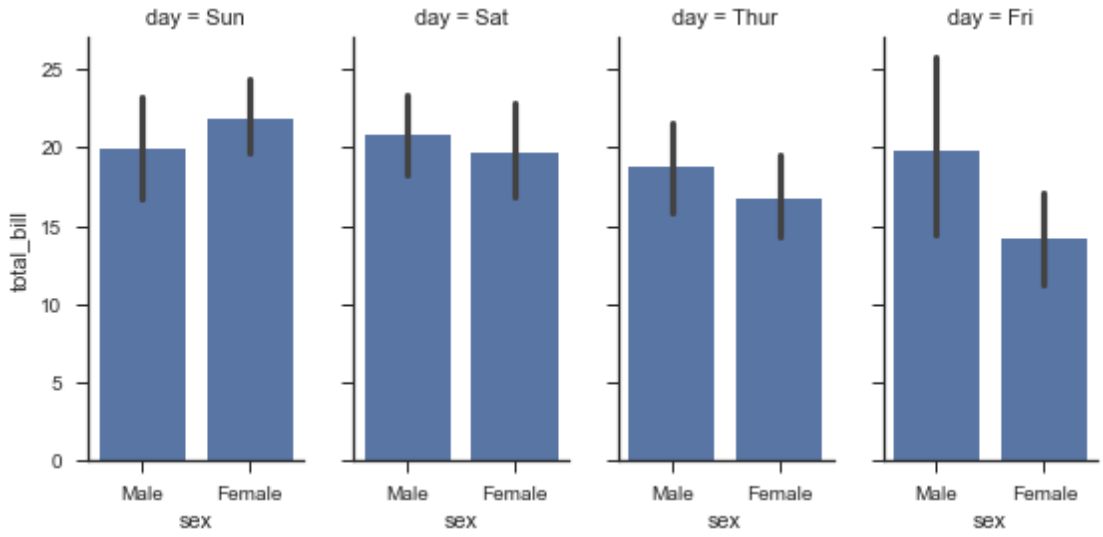

```
In [79]: g = sns.FacetGrid(tips, col="sex", hue="smoker")
         g.map(plt.scatter, "total_bill", "tip", alpha=.7)
         g.add_legend();
```




```
In [80]: g = sns.FacetGrid(tips, row="smoker", col="time", margin_titles=True)
         g.map(sns.regplot, "size", "total_bill", color=".1", fit_reg=False, x_jitter=.1);
```

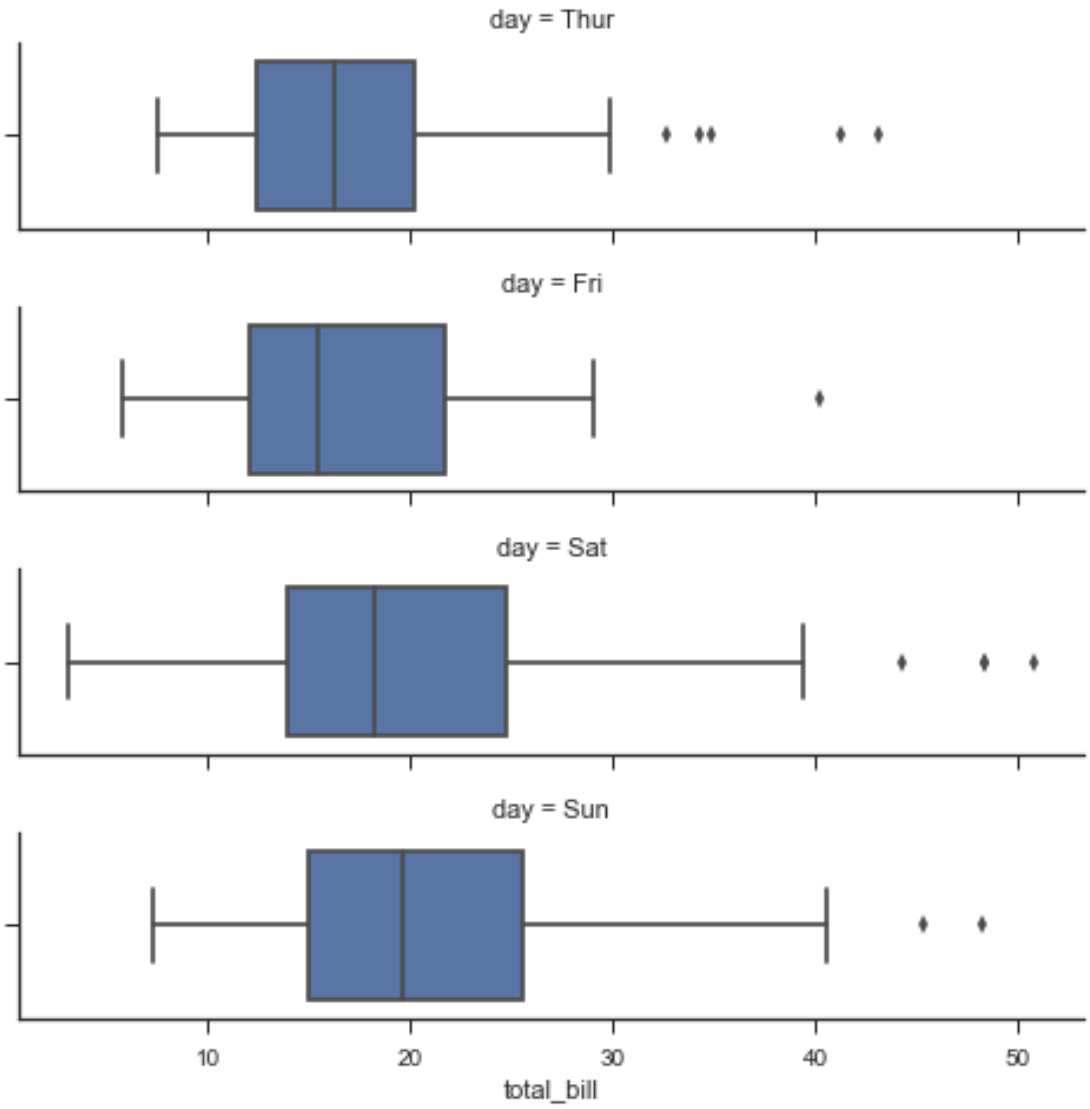



```
In [81]: g = sns.FacetGrid(tips, col="day", size=4, aspect=.5)
         g.map(sns.barplot, "sex", "total_bill");
```

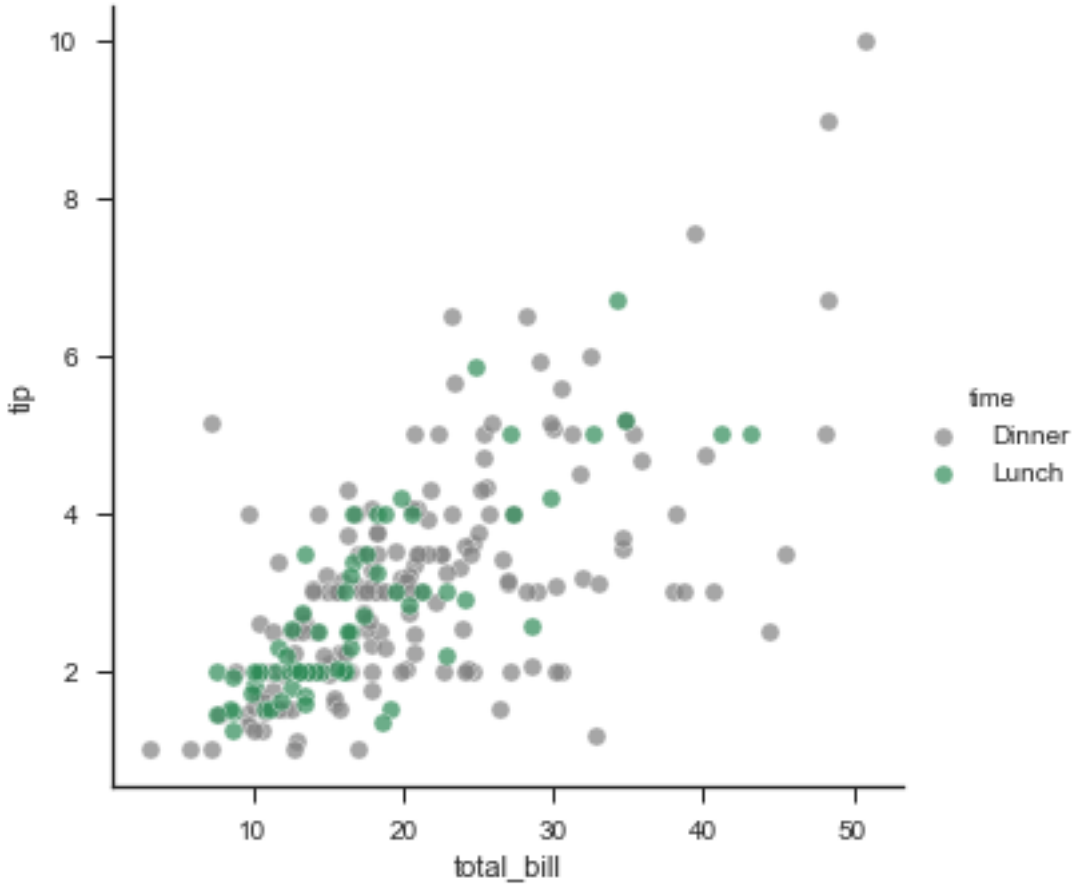



```
In [82]: from pandas import Categorical
         ordered_days = tips.day.value_counts().index
         print (ordered_days)
         ordered_days = Categorical(['Thur', 'Fri', 'Sat', 'Sun'])
         g = sns.FacetGrid(tips, row="day", row_order=ordered_days,
                           size=1.7, aspect=4,)
         g.map(sns.boxplot, "total_bill");
```

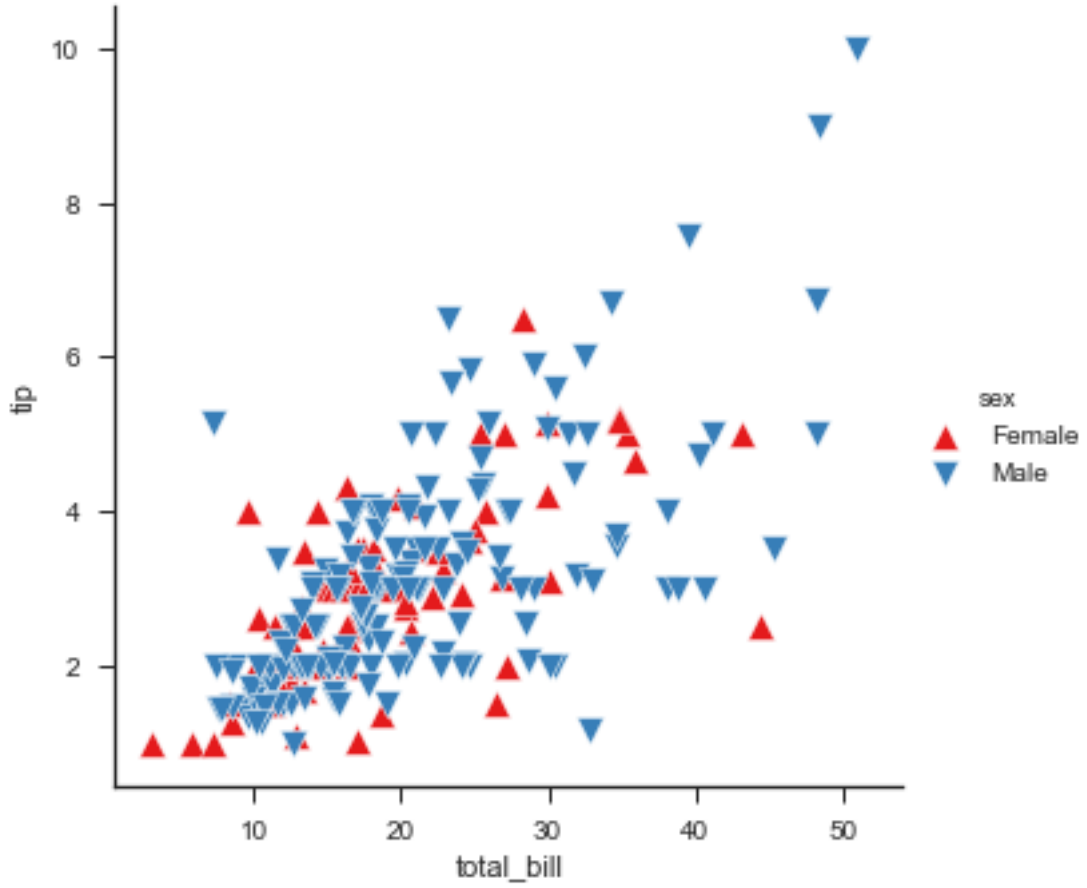
```
Index(['Sat', 'Sun', 'Thur', 'Fri'], dtype='object')
```



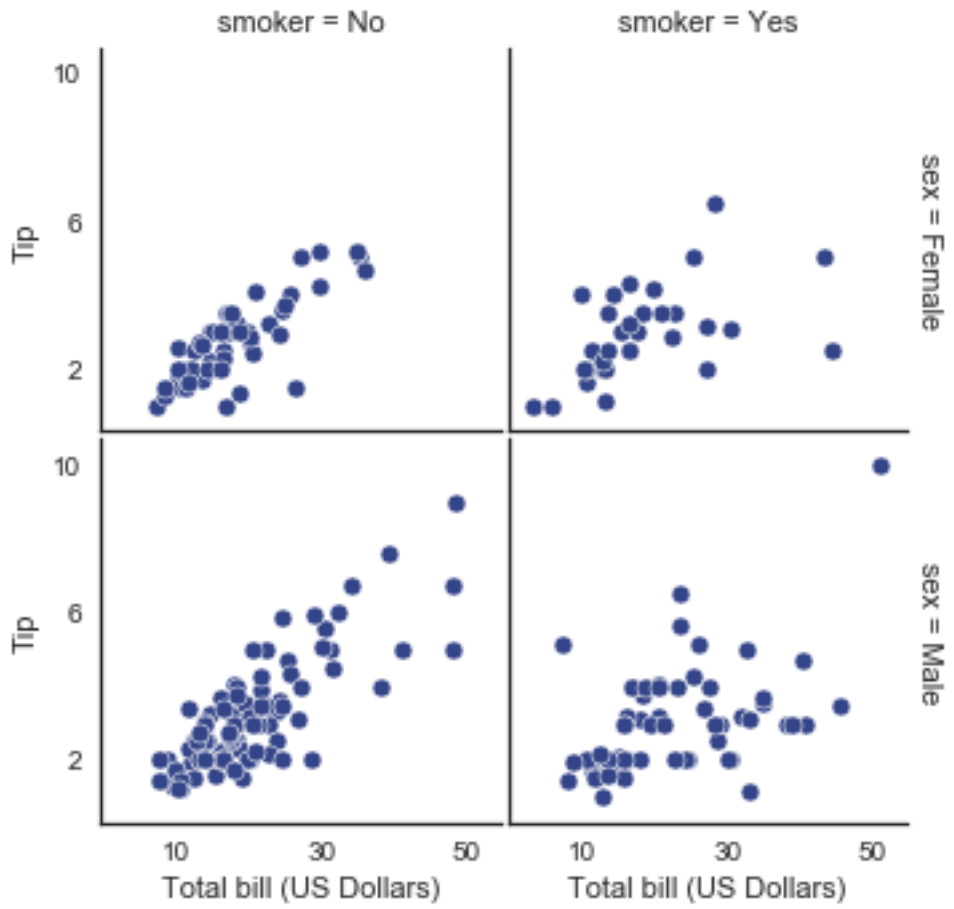

```
In [83]: pal = dict(Lunch="seagreen", Dinner="gray")
         g = sns.FacetGrid(tips, hue="time", palette=pal, size=5)
         g.map(plt.scatter, "total_bill", "tip", s=50, alpha=.7, linewidth=.5, edgecolor="white")
         g.add_legend();
```



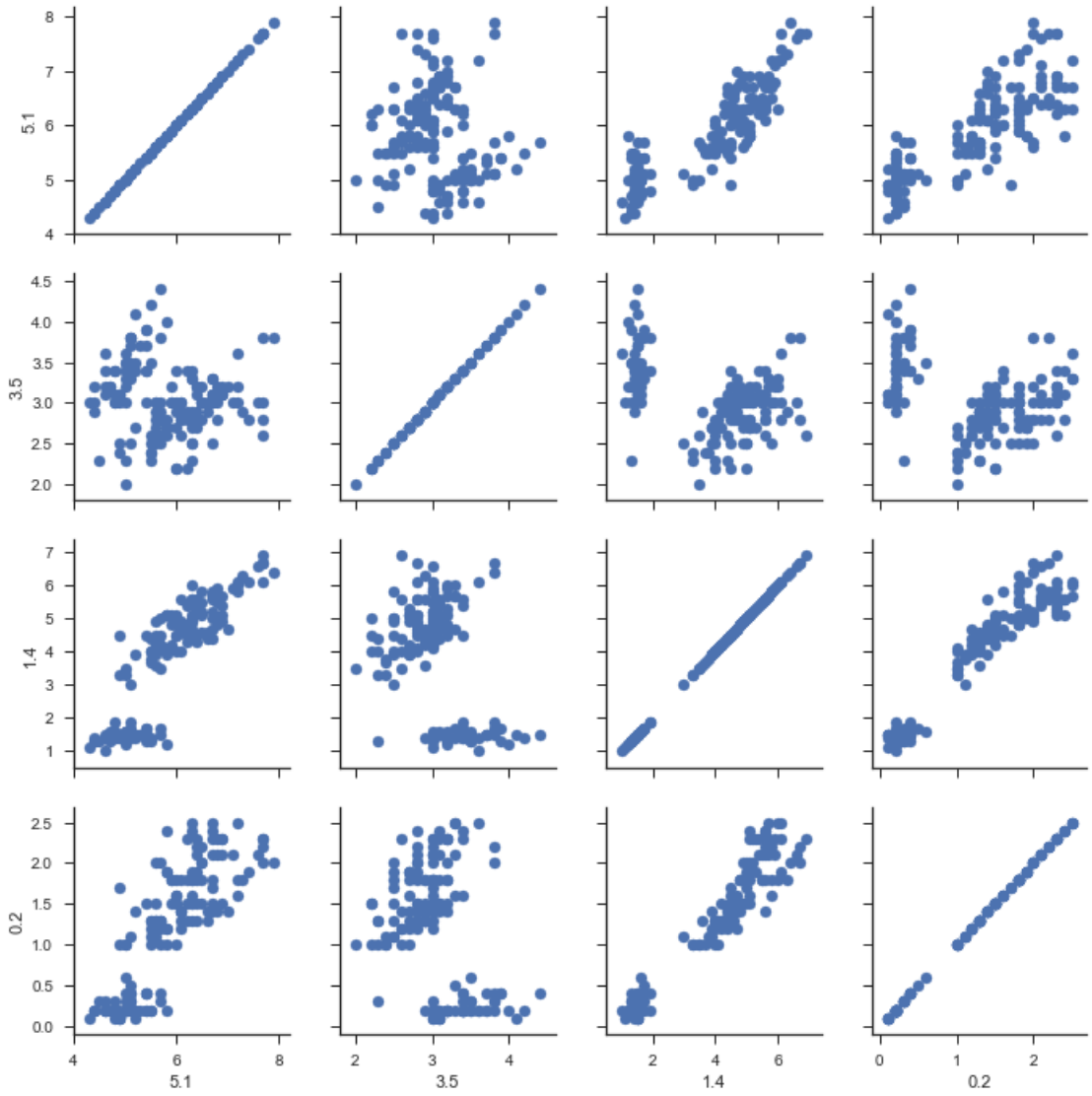

```
In [84]: g = sns.FacetGrid(tips, hue="sex", palette="Set1", size=5, hue_kws={"marker": ["^", "v"]}
         g.map(plt.scatter, "total_bill", "tip", s=100, linewidth=.5, edgecolor="white")
         g.add_legend();
```



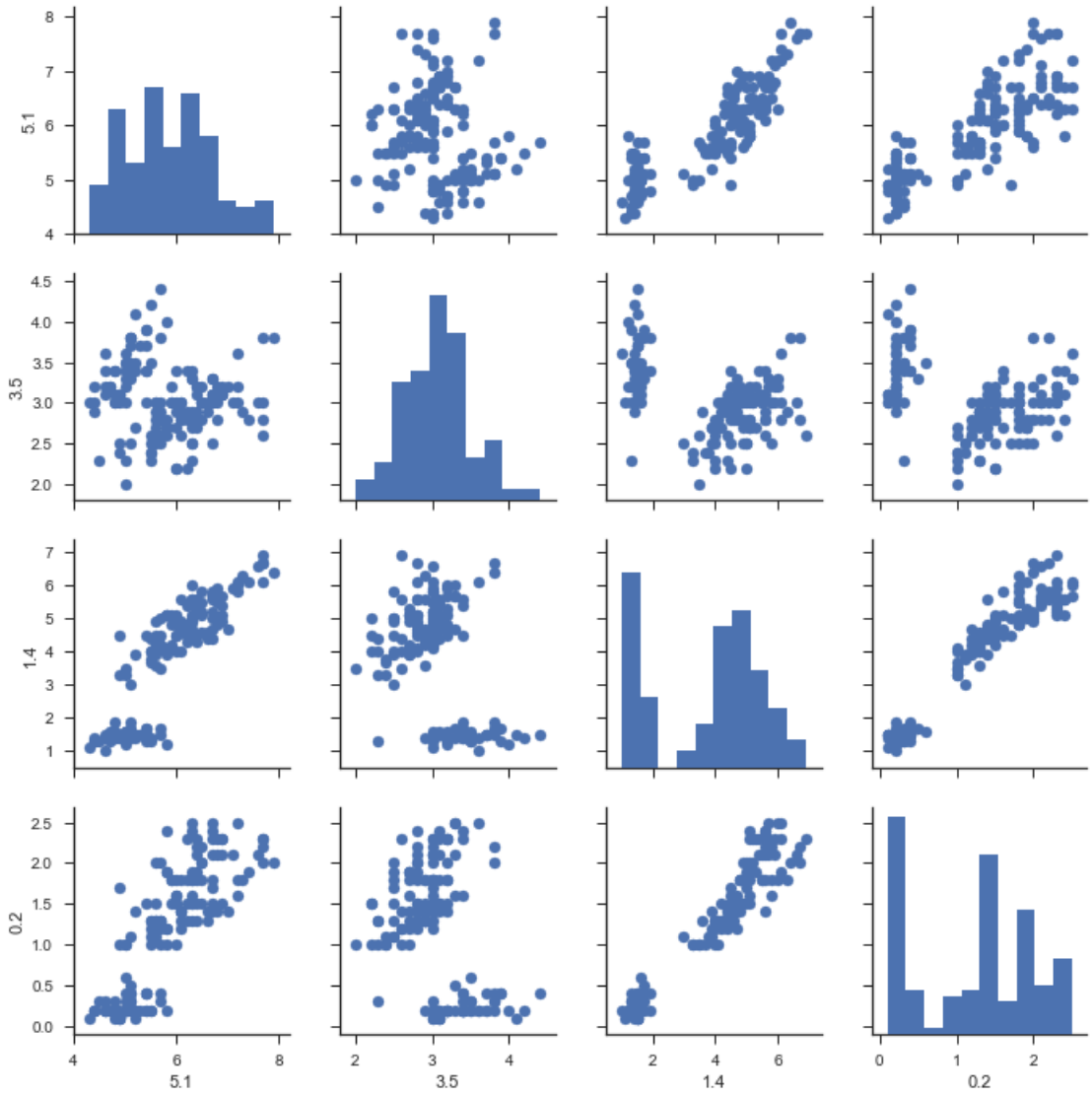

```
In [85]: with sns.axes_style("white"):
             g = sns.FacetGrid(tips, row="sex", col="smoker", margin_titles=True, size=2.5)
         g.map(plt.scatter, "total_bill", "tip", color="#334488", edgecolor="white", lw=.5);
         g.set_axis_labels("Total bill (US Dollars)", "Tip");
         g.set(xticks=[10, 30, 50], yticks=[2, 6, 10]);
         g.fig.subplots_adjust(wspace=.02, hspace=.02);
         #g.fig.subplots_adjust(left  = 0.125,right = 0.5,bottom = 0.1,top = 0.9, wspace=.02, hspa
```



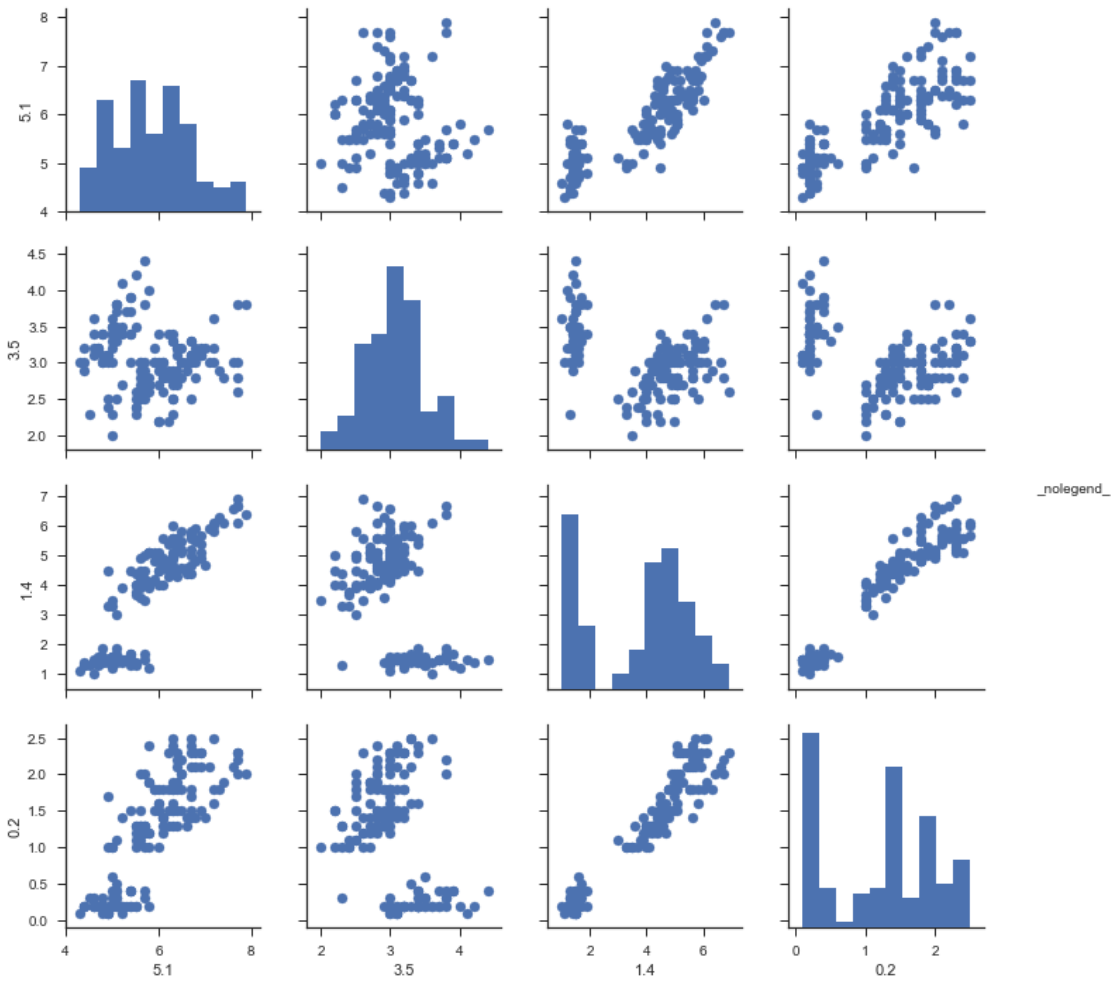

```
In [86]: g = sns.PairGrid(iris)
         g.map(plt.scatter);
```




```
In [87]: g = sns.PairGrid(iris)
         g.map_diag(plt.hist)
         g.map_offdiag(plt.scatter);
```



In [88]:
```
#g = sns.PairGrid(iris, hue="Species")# 这句跑不出来，没有 Species 这个参数
g = sns.PairGrid(iris)

g.map_diag(plt.hist)
g.map_offdiag(plt.scatter)
g.add_legend();

#help(sns.PairGrid)
```

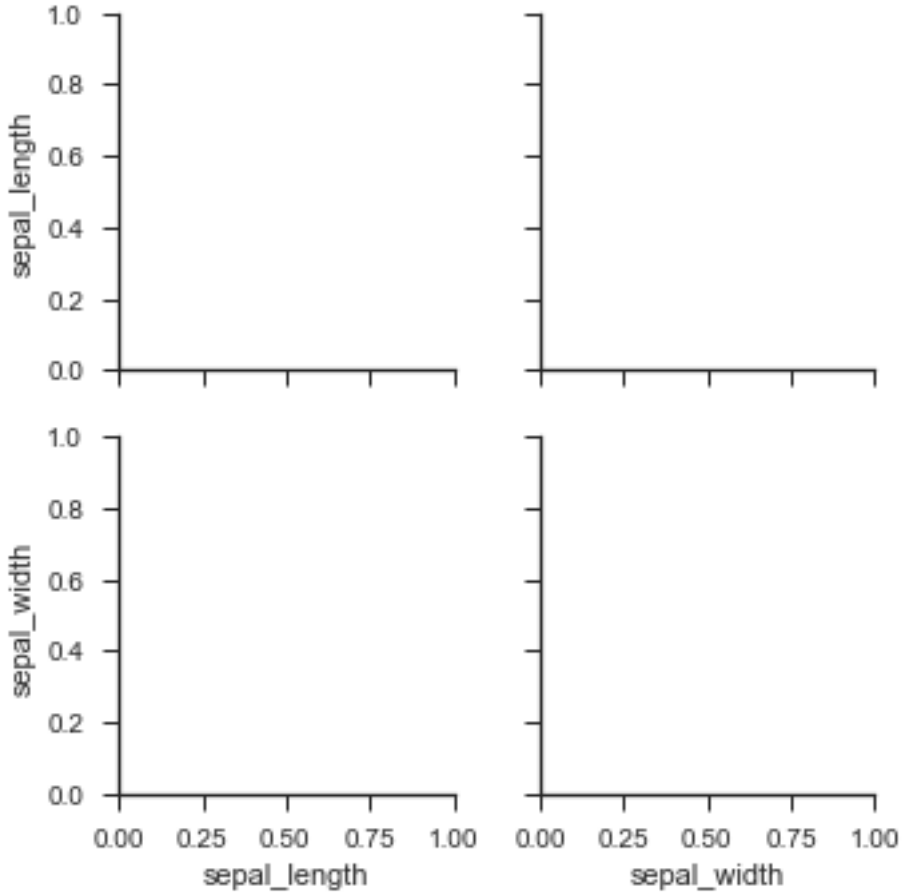



```
In [89]: #g = sns.PairGrid(iris, vars=["sepal_length", "sepal_width"],hue="species")
         g = sns.PairGrid(iris, vars=["sepal_length", "sepal_width"] )# 这句也卡了。。。不明原因

         g.map(plt.scatter)

         ---------------------------------------------------------------------------

         KeyError                                  Traceback (most recent call last)

         D:\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, t
      2392             try:
      -> 2393                 return self._engine.get_loc(key)
      2394             except KeyError:
```

```
    pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc (pandas\_libs\index.c:523

    pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc (pandas\_libs\index.c:508

    pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_it

    pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_it

    KeyError: 'sepal_length'


During handling of the above exception, another exception occurred:

    KeyError                                  Traceback (most recent call last)

    <ipython-input-89-e2f4ee0e524d> in <module>()
      2 g = sns.PairGrid(iris, vars=["sepal_length", "sepal_width"] )# 这句也卡了。。。不明原因
      3
----> 4 g.map(plt.scatter)


    D:\Anaconda3\lib\site-packages\seaborn\axisgrid.py in map(self, func, **kwargs)
   1286
   1287                     color = self.palette[k] if kw_color is None else kw_color
-> 1288                     func(data_k[x_var], data_k[y_var],
   1289                          label=label_k, color=color, **kwargs)
   1290


    D:\Anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   2060             return self._getitem_multilevel(key)
   2061         else:
```

```
-> 2062             return self._getitem_column(key)
   2063
   2064     def _getitem_column(self, key):


    D:\Anaconda3\lib\site-packages\pandas\core\frame.py in _getitem_column(self, key)
   2067         # get column
   2068         if self.columns.is_unique:
-> 2069             return self._get_item_cache(key)
   2070
   2071         # duplicate columns & possible reduce dimensionality


    D:\Anaconda3\lib\site-packages\pandas\core\generic.py in _get_item_cache(self, item)
   1532         res = cache.get(item)
   1533         if res is None:
-> 1534             values = self._data.get(item)
   1535             res = self._box_item_values(item, values)
   1536             cache[item] = res


    D:\Anaconda3\lib\site-packages\pandas\core\internals.py in get(self, item, fastpath)
   3588
   3589             if not isnull(item):
-> 3590                 loc = self.items.get_loc(item)
   3591             else:
   3592                 indexer = np.arange(len(self.items))[isnull(self.items)]


    D:\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, t
   2393                 return self._engine.get_loc(key)
   2394             except KeyError:
-> 2395                 return self._engine.get_loc(self._maybe_cast_indexer(key))
   2396
   2397         indexer = self.get_indexer([key], method=method, tolerance=tolerance)


    pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc (pandas\_libs\index.c:523
```

```
pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc (pandas\_libs\index.c:508

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_it

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_it

KeyError: 'sepal_length'
```



In [ ]:
```
g = sns.PairGrid(tips, hue="size", palette="GnBu_d")
g.map(plt.scatter, s=50, edgecolor="white")
g.add_legend();
```

```
In [ ]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np;
        np.random.seed(0)
        import seaborn as sns;
        sns.set()

In [ ]: uniform_data = np.random.rand(3, 3)
        print (uniform_data)
        heatmap = sns.heatmap(uniform_data)

In [ ]: ax = sns.heatmap(uniform_data, vmin=0.2, vmax=0.5)

In [ ]: normal_data = np.random.randn(3, 3)
        print (normal_data)
        ax = sns.heatmap(normal_data, center=0)

In [ ]: #flights = sns.load_dataset("flights")
        flights = pd.read_csv("flights.csv")
        flights.head()

In [ ]: flights = flights.pivot("month", "year", "passengers")
        print (flights)
        ax = sns.heatmap(flights)

In [ ]: ax = sns.heatmap(flights, annot=True,fmt="d")

In [ ]: ax = sns.heatmap(flights, linewidths=.5)

In [ ]: ax = sns.heatmap(flights, cmap="YlGnBu")

In [ ]: ax = sns.heatmap(flights, cbar=False)

In [ ]:
```