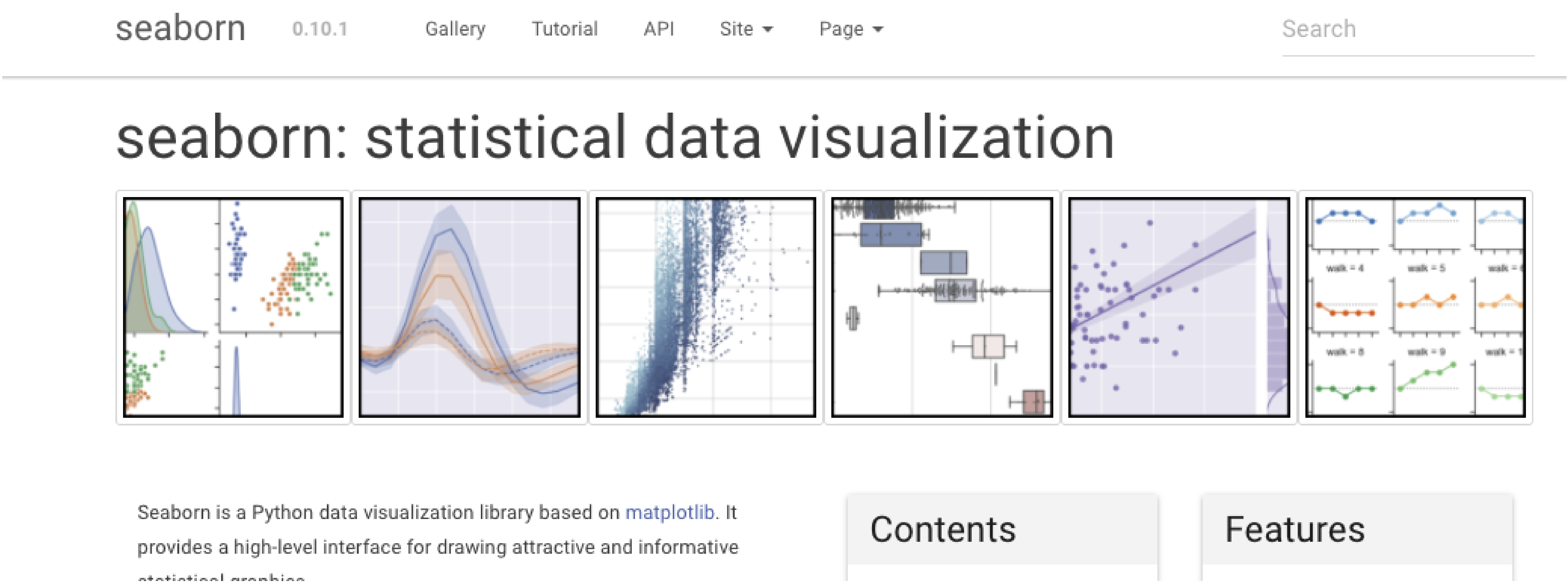
Visualizing regressions

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

Seaborn



seaborn 0.10.1 Gallery Tutorial API Site ▾ Page ▾ Search

seaborn: statistical data visualization

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics

Contents

Features

http://seaborn.pydata.org/

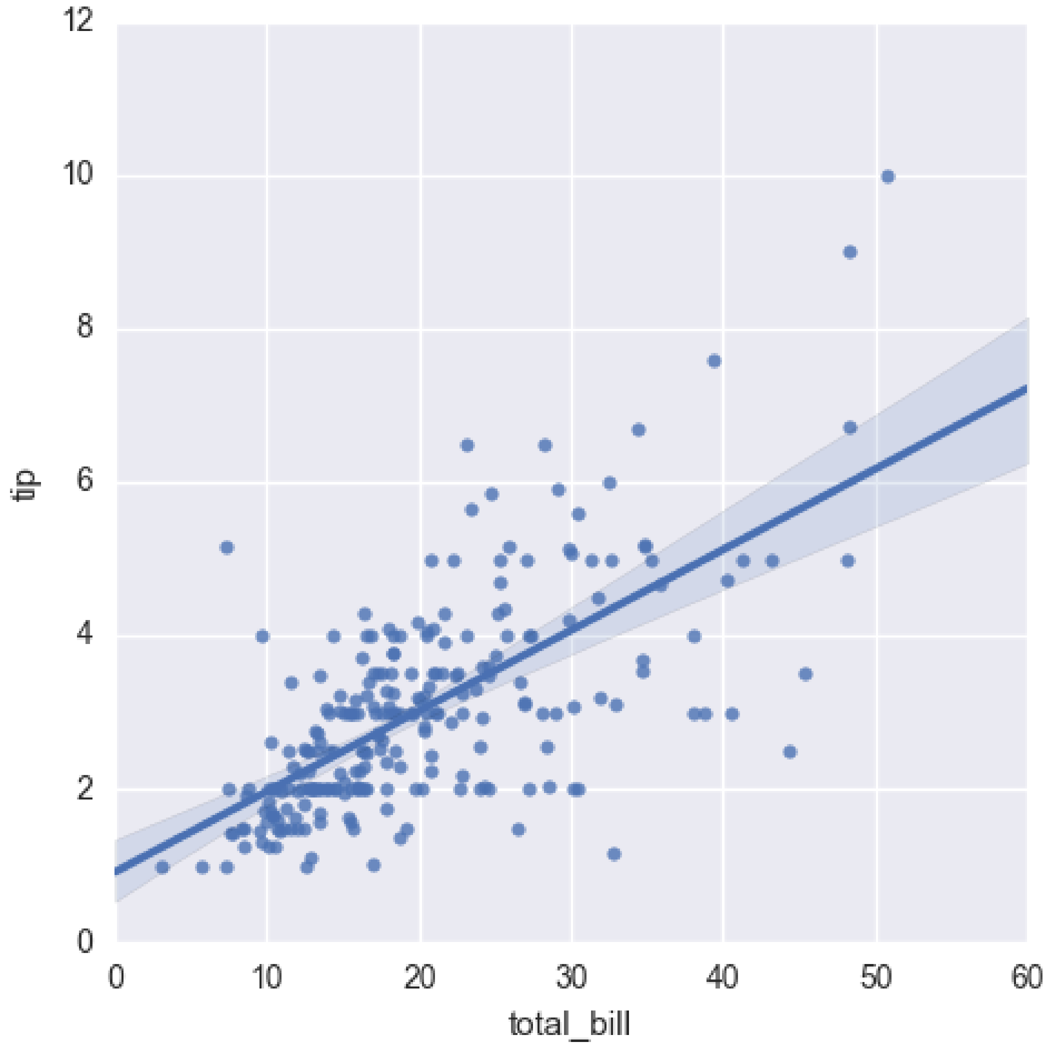
Recap: pandas DataFrames

- Labelled tabular data structure
- Labels on rows: `index`
- Labels on columns: `columns`
- Columns are `pandas` `Series`

Tips DataFrame

	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	3.5	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
...	...	...	...	...	...	...	...

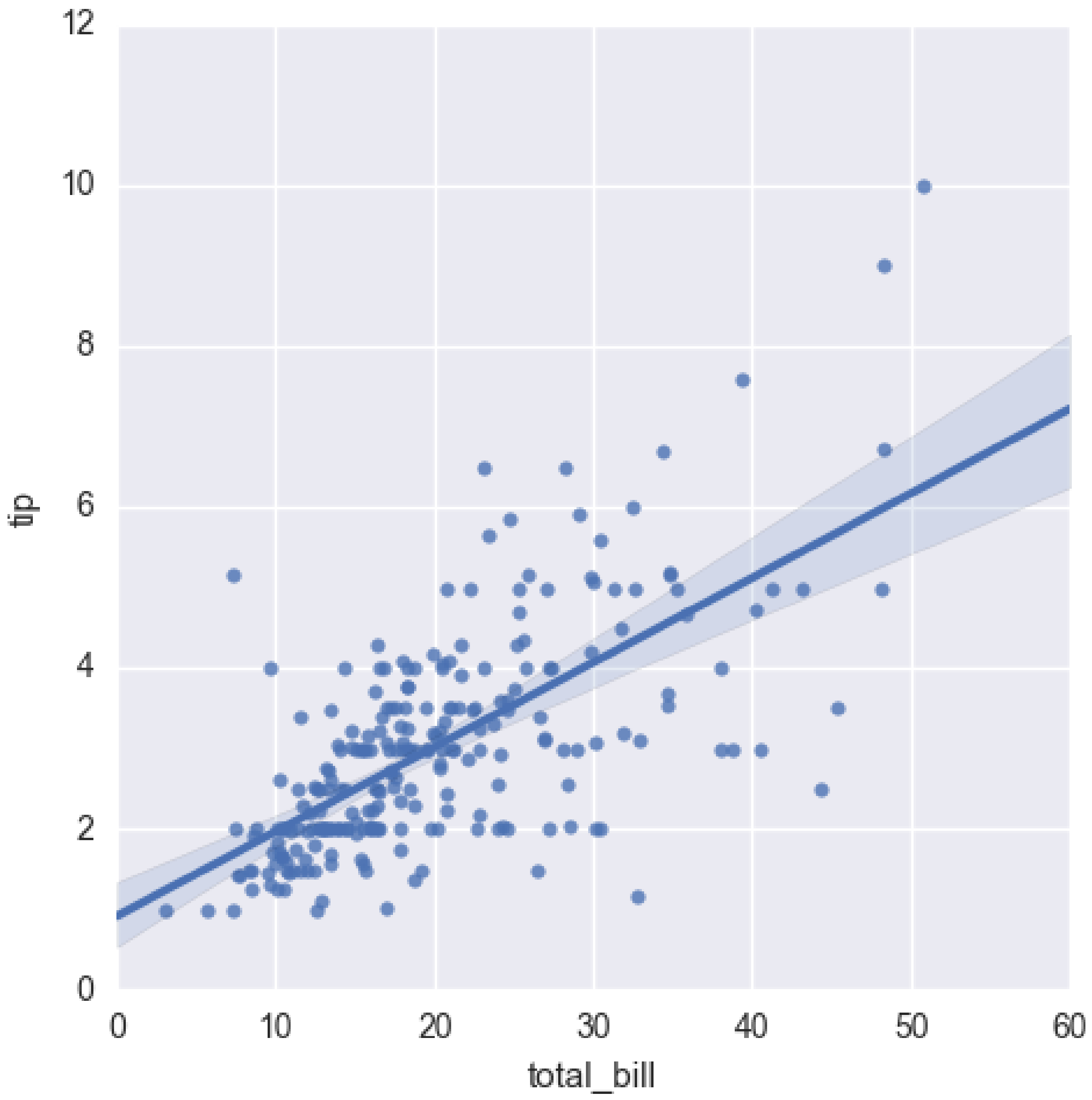
Linear regression plots



- 95% confidence interval highlighted

Using lmplot()

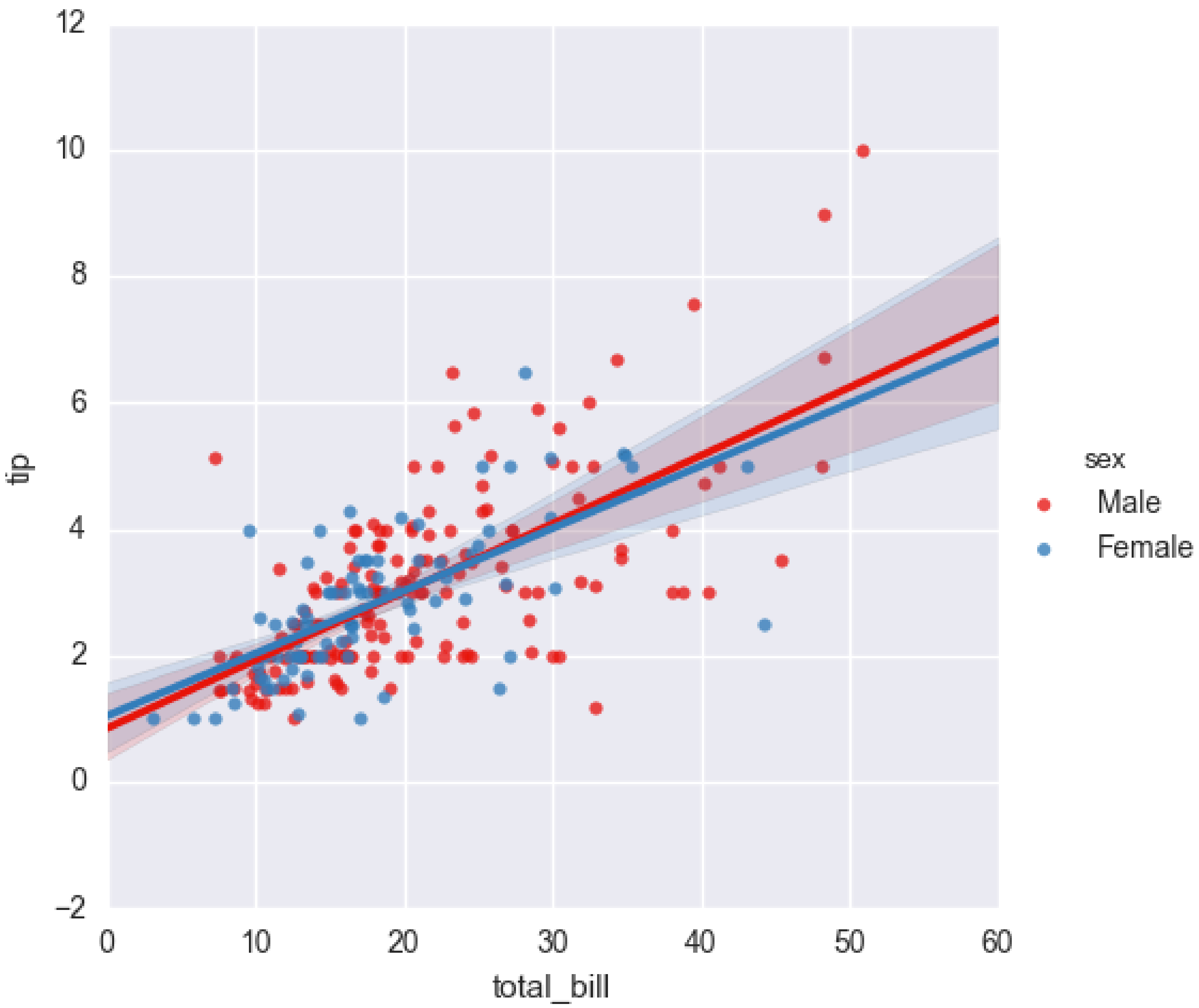
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
tips = sns.load_dataset('tips')
sns.lmplot(x='total_bill',
           y='tip',
           data=tips)
plt.show()
```



Factors

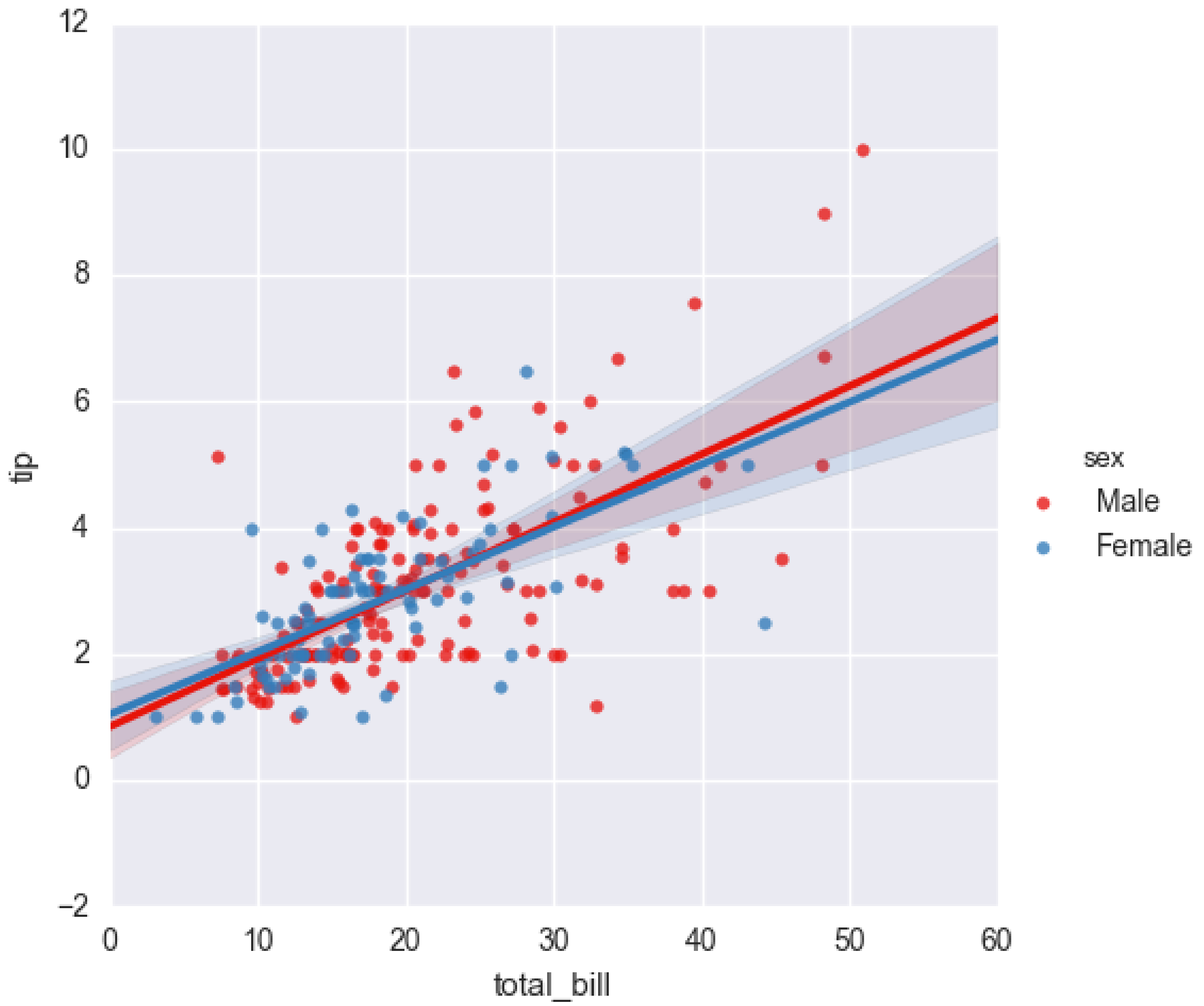
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	***Female***	No	Sun	Dinner	2
1	10.34	1.66	***Male***	No	Sun	Dinner	3
2	21.01	3.5	***Male***	No	Sun	Dinner	3
3	23.68	3.31	***Male***	No	Sun	Dinner	2
4	24.59	3.61	***Female***	No	Sun	Dinner	4
...	...	...	...	...	...	...	...

Grouping factors (same plot)

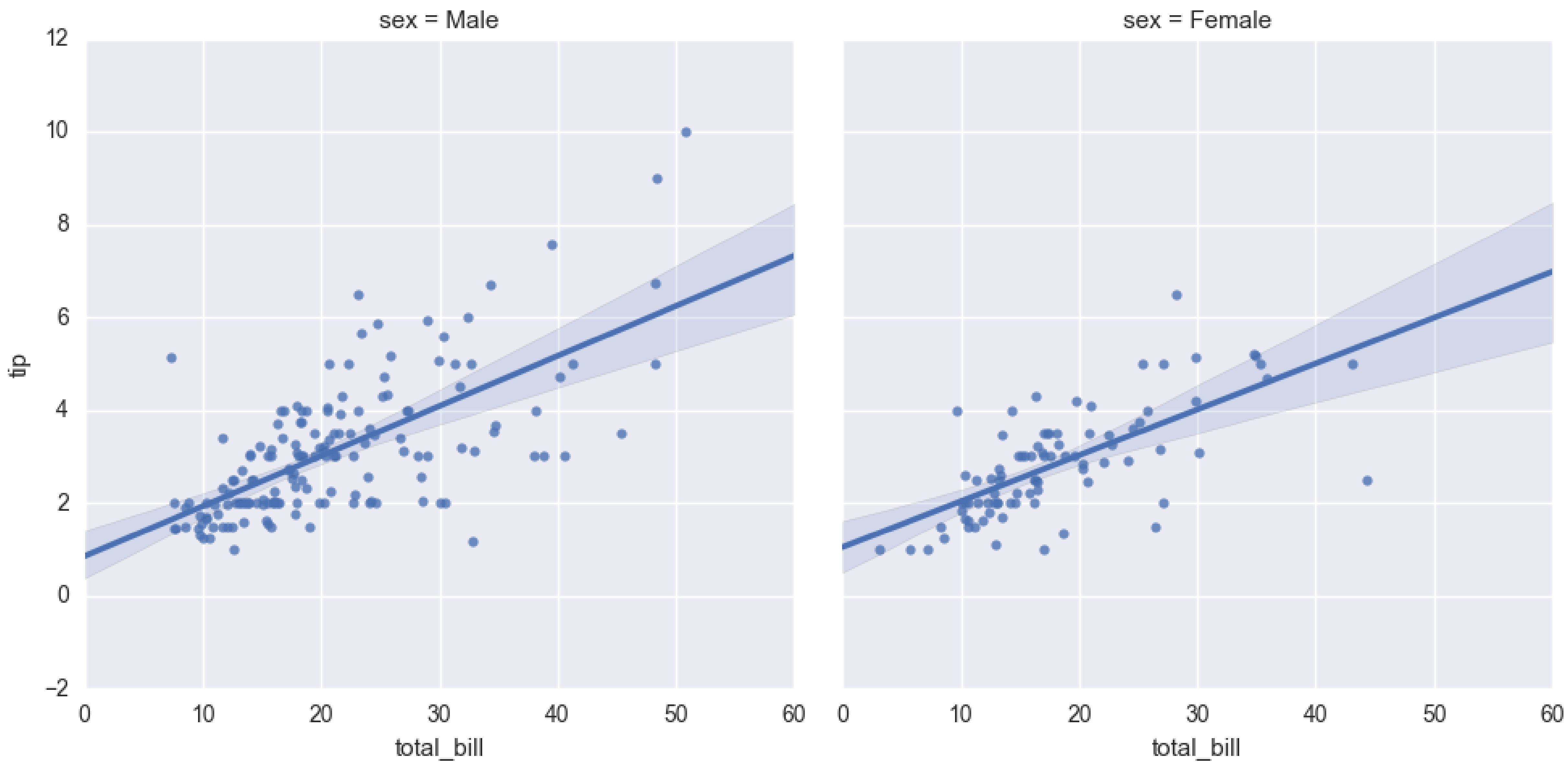


Using hue

```
sns.lmplot(x='total_bill', y='tip',
           data=tips,
           hue='sex',
           palette='Set1')
plt.show()
```

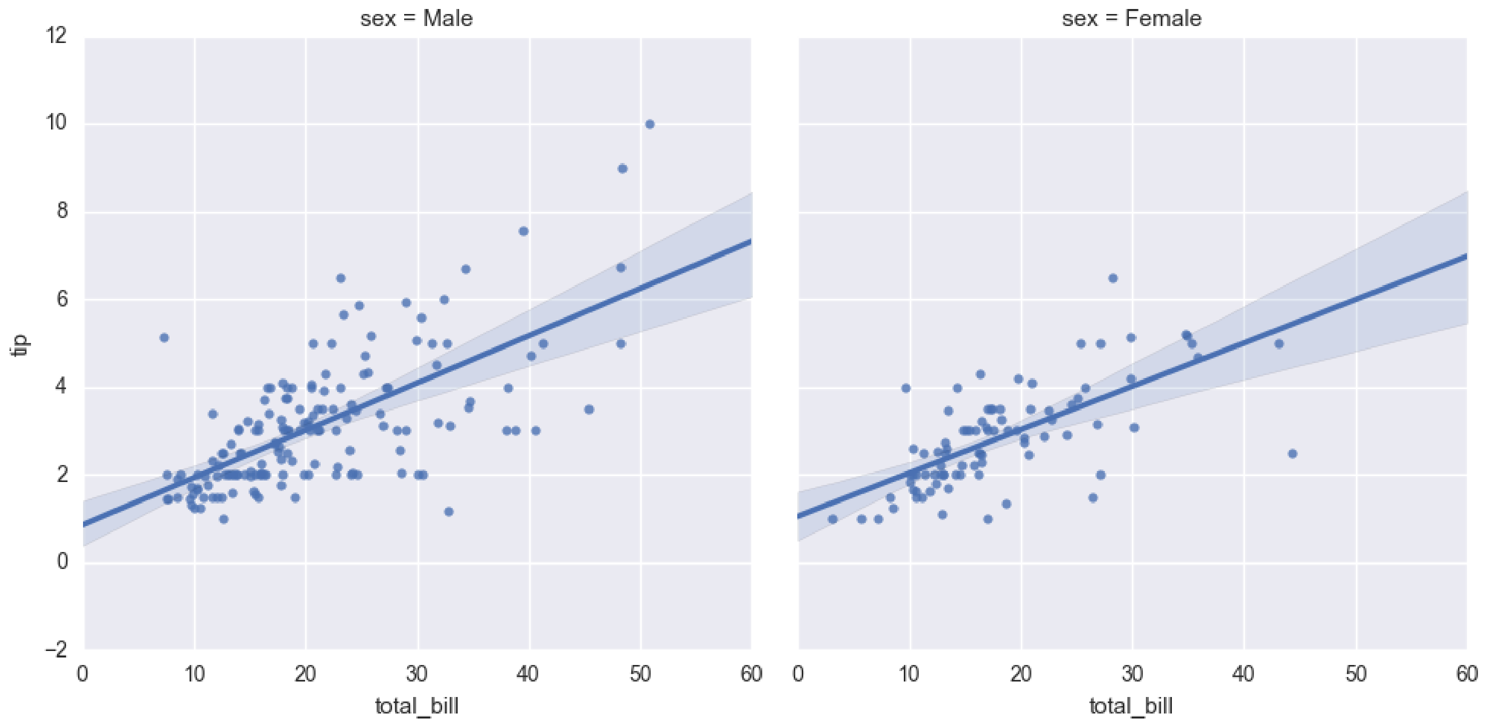


Grouping factors (subplots)

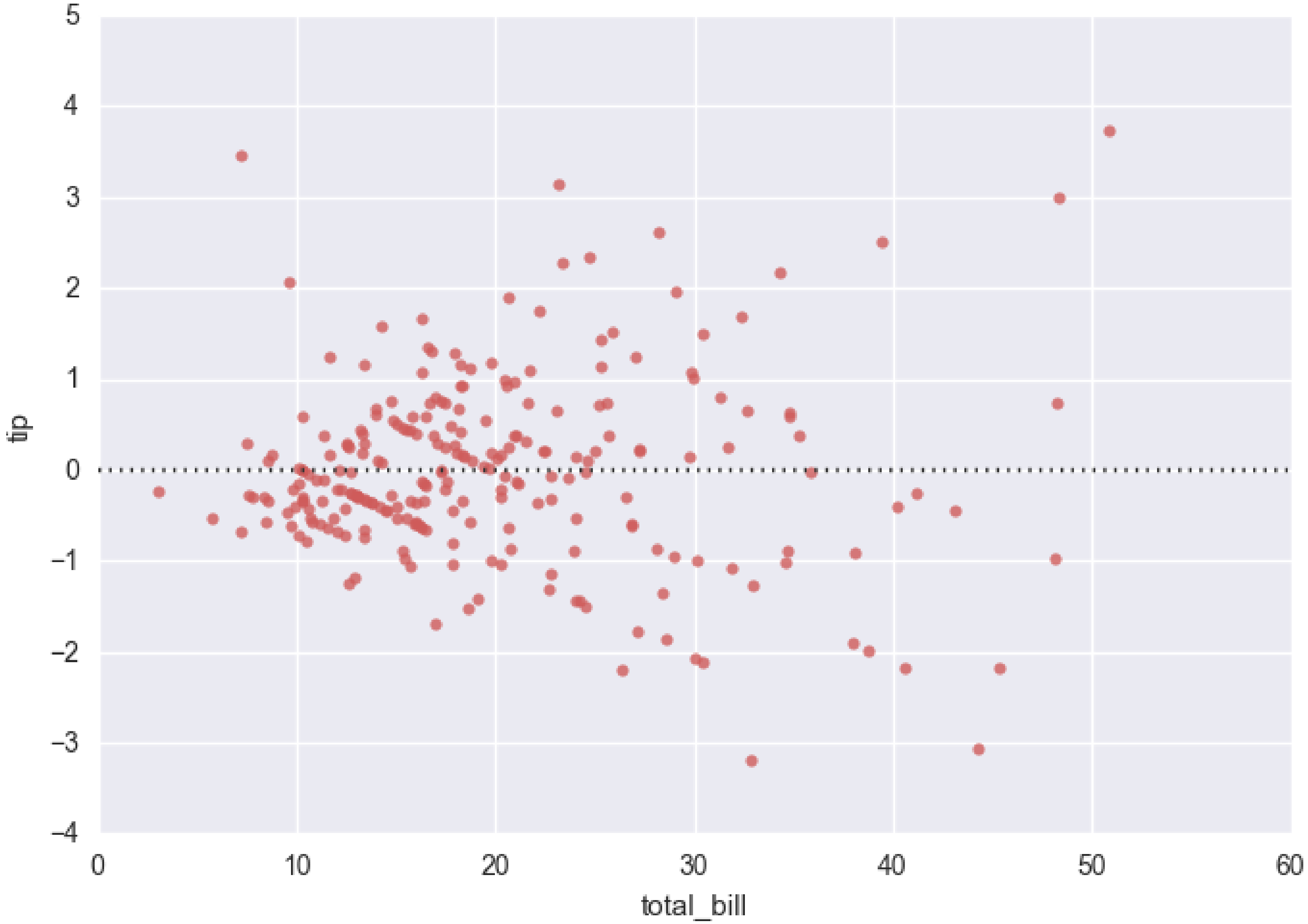


Using col

```
sns.lmplot(x='total_bill', y='tip',
           data=tips,
           col='sex')
plt.show()
```



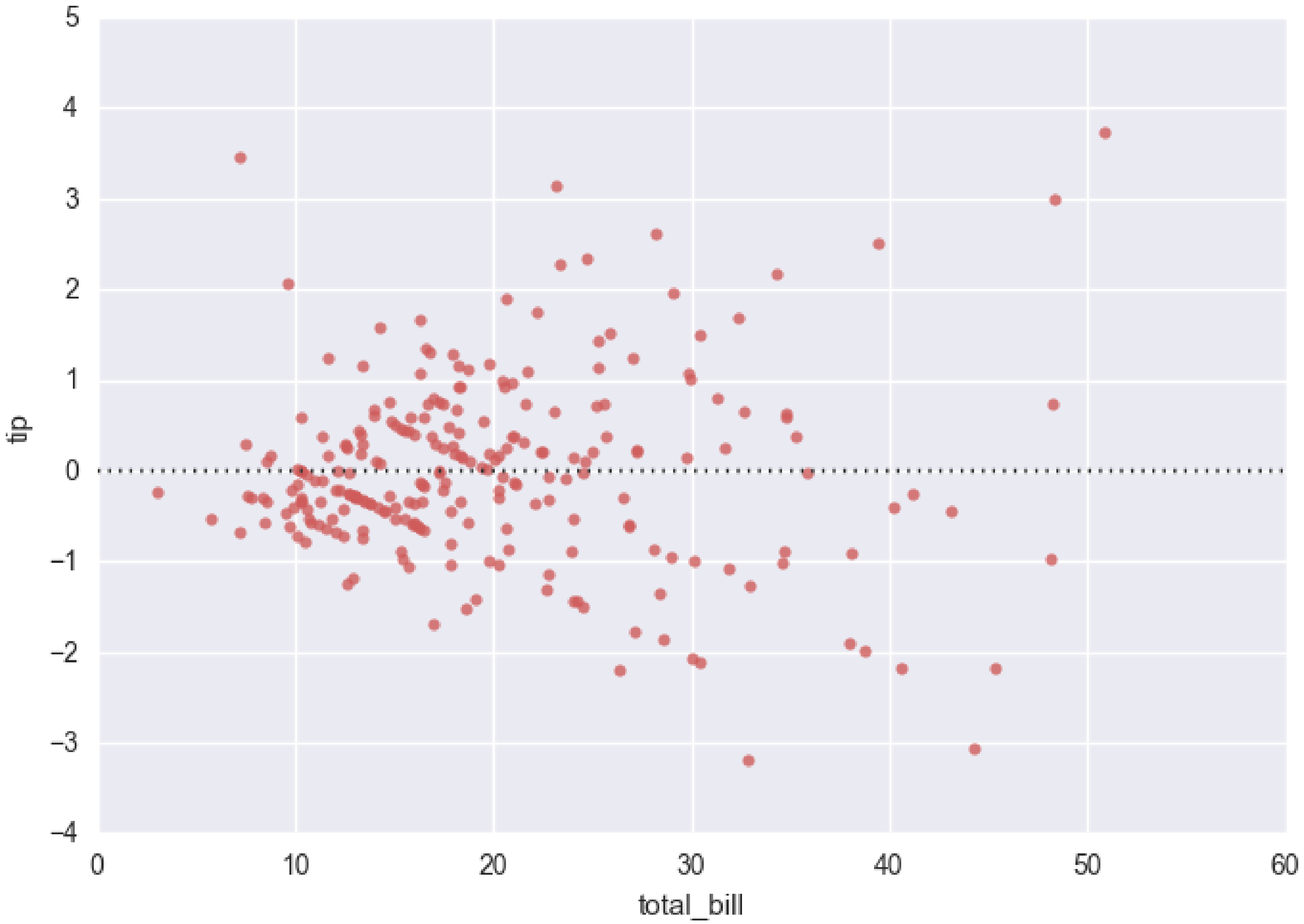
Residual plots



Using residplot()

```
sns.residplot(x='age',y='fare',
              data=tips,
              color='indianred')
plt.show()
```

- Similar arguments as `lmplot()` but more flexible
- `x` , `y` can be arrays or strings
- `data` is DataFrame (optional)
- Optional arguments (e.g., `color`) as in `matplotlib`



Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Visualizing univariate distributions

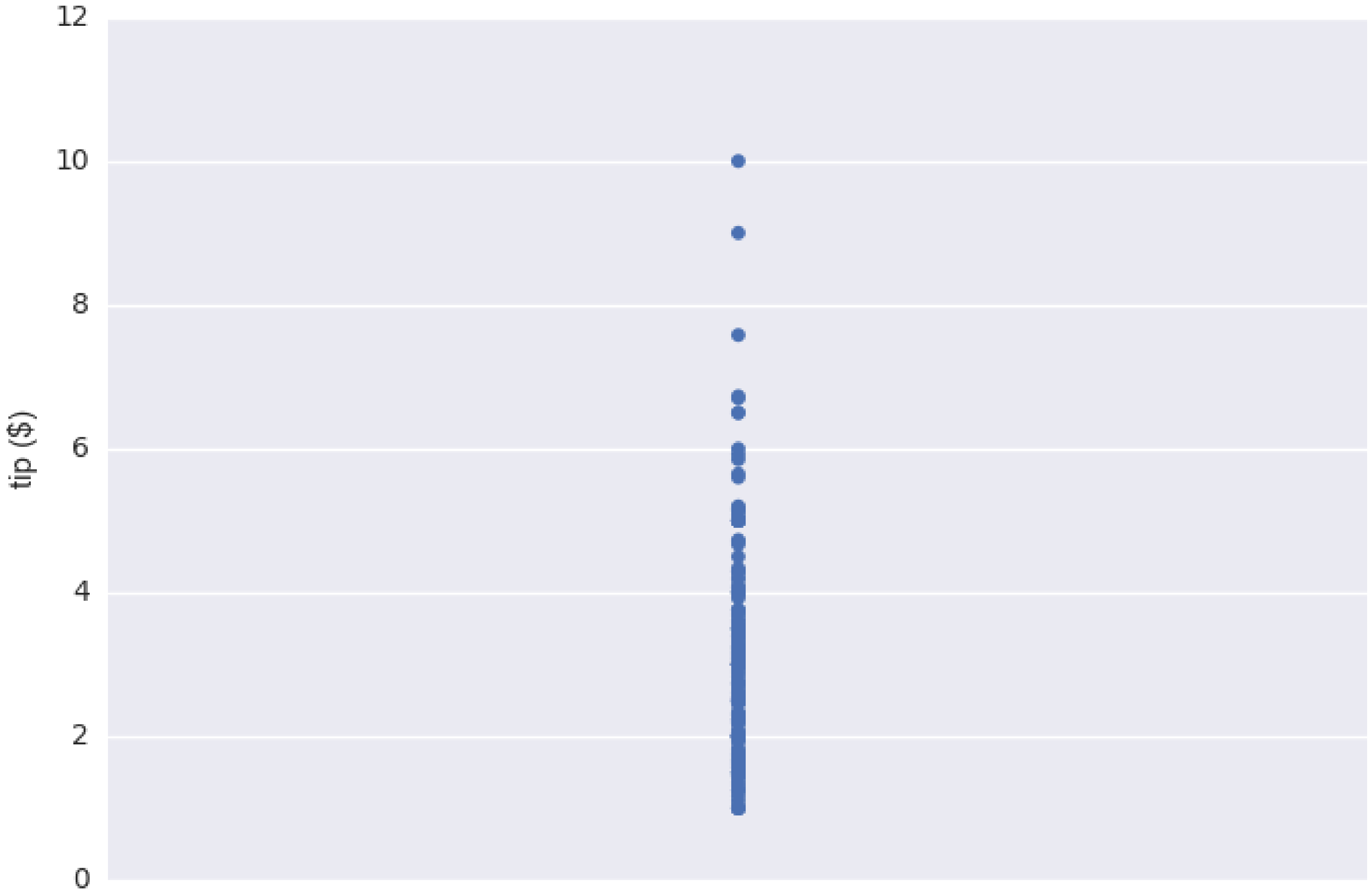
INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

Visualizing data

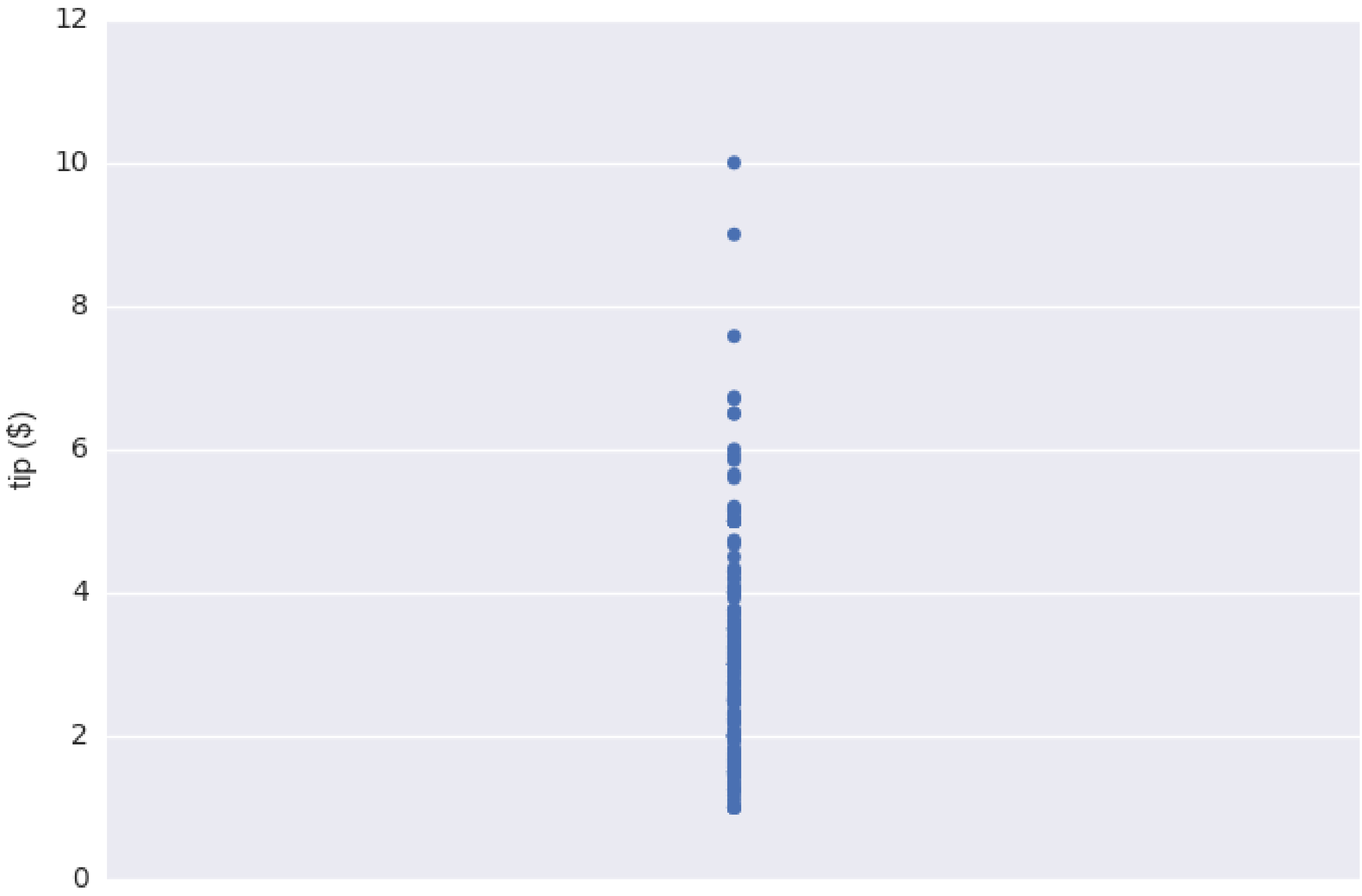
- Univariate → "one variable"
- Visualization techniques for sampled univariate data
  - Strip plots
  - Swarm plots
  - Violin plots

Strip plot



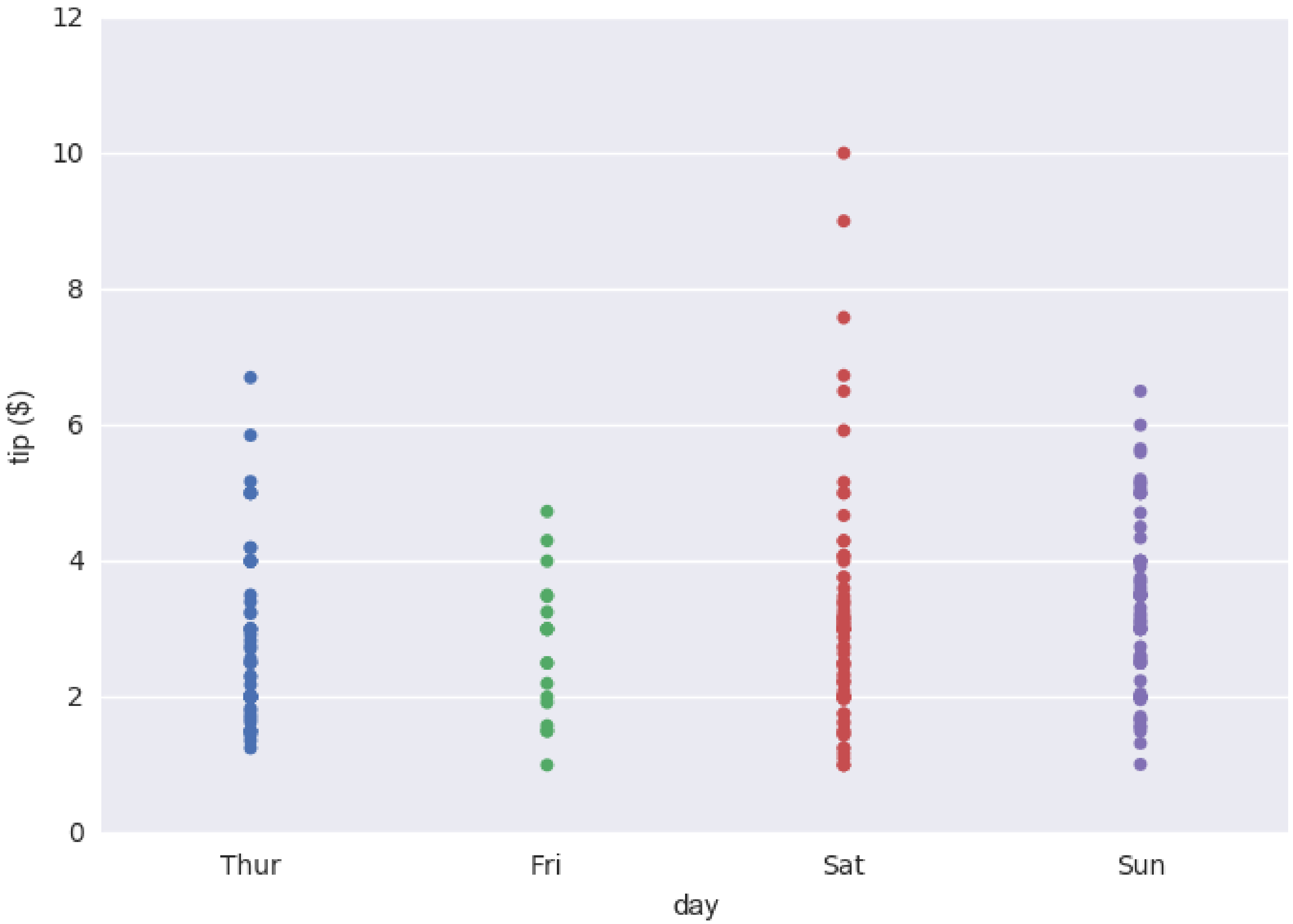
Using stripplot()

```
sns.stripplot(y='tip', data=tips)
plt.ylabel('tip ($)')
plt.show()
```



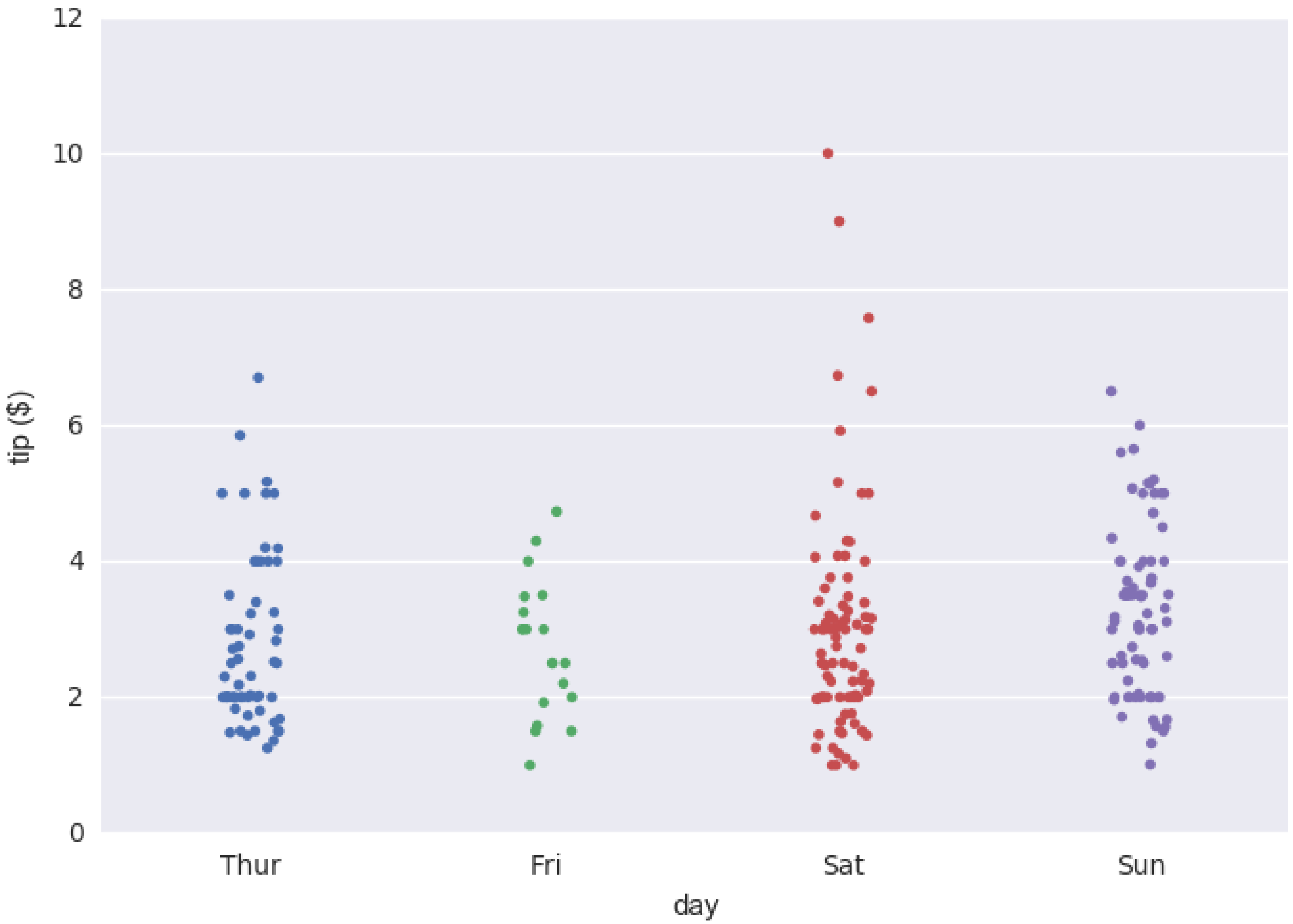
Grouping with stripplot()

```
sns.stripplot(x='day',
              y='tip', data=tip)
plt.ylabel('tip ($)')
plt.show()
```

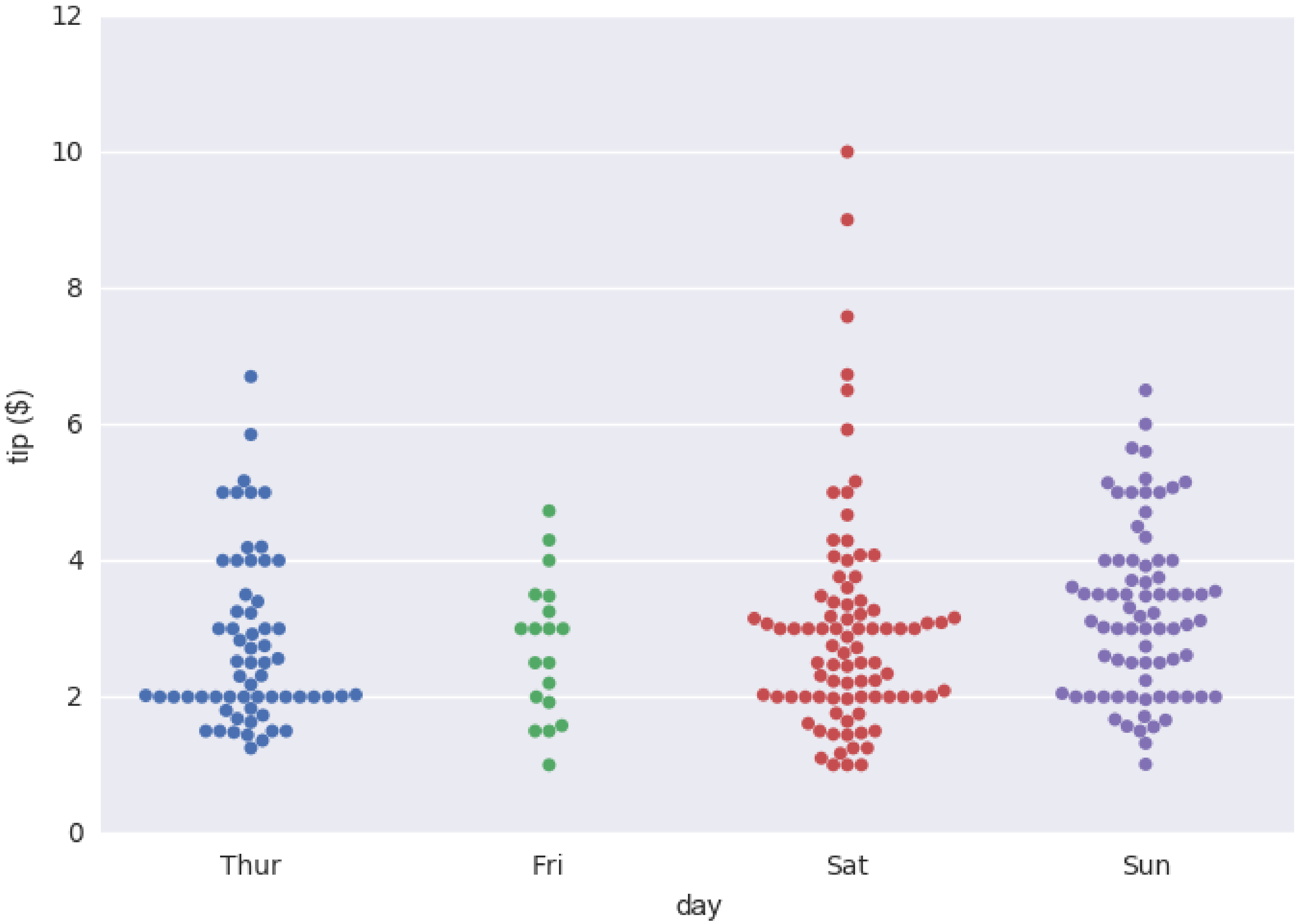


Spreading out strip plots

```
sns.stripplot(x='day',
              y='tip', data=tip,
              size=4,
              jitter=True)
plt.ylabel('tip ($)')
plt.show()
```

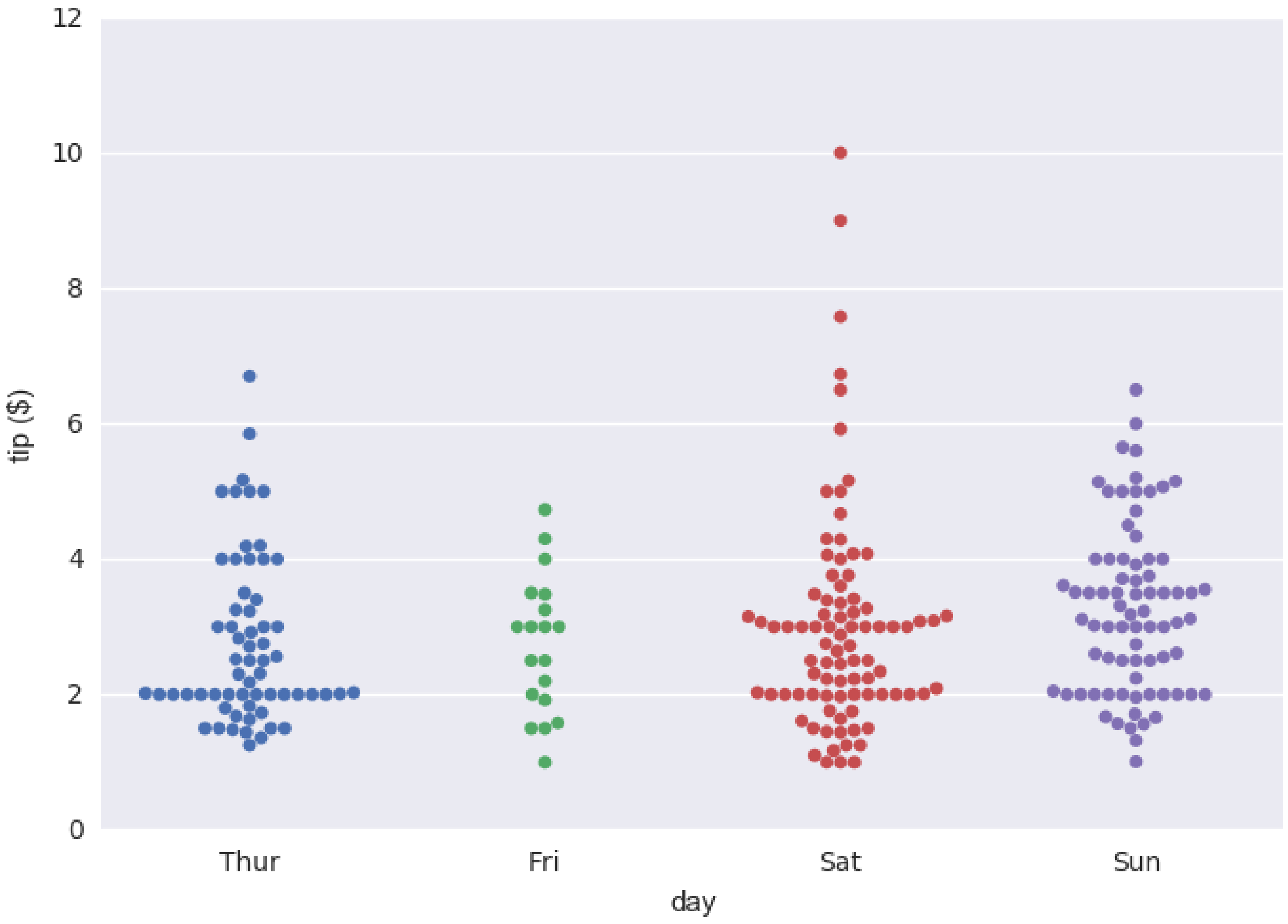


Swarm plot

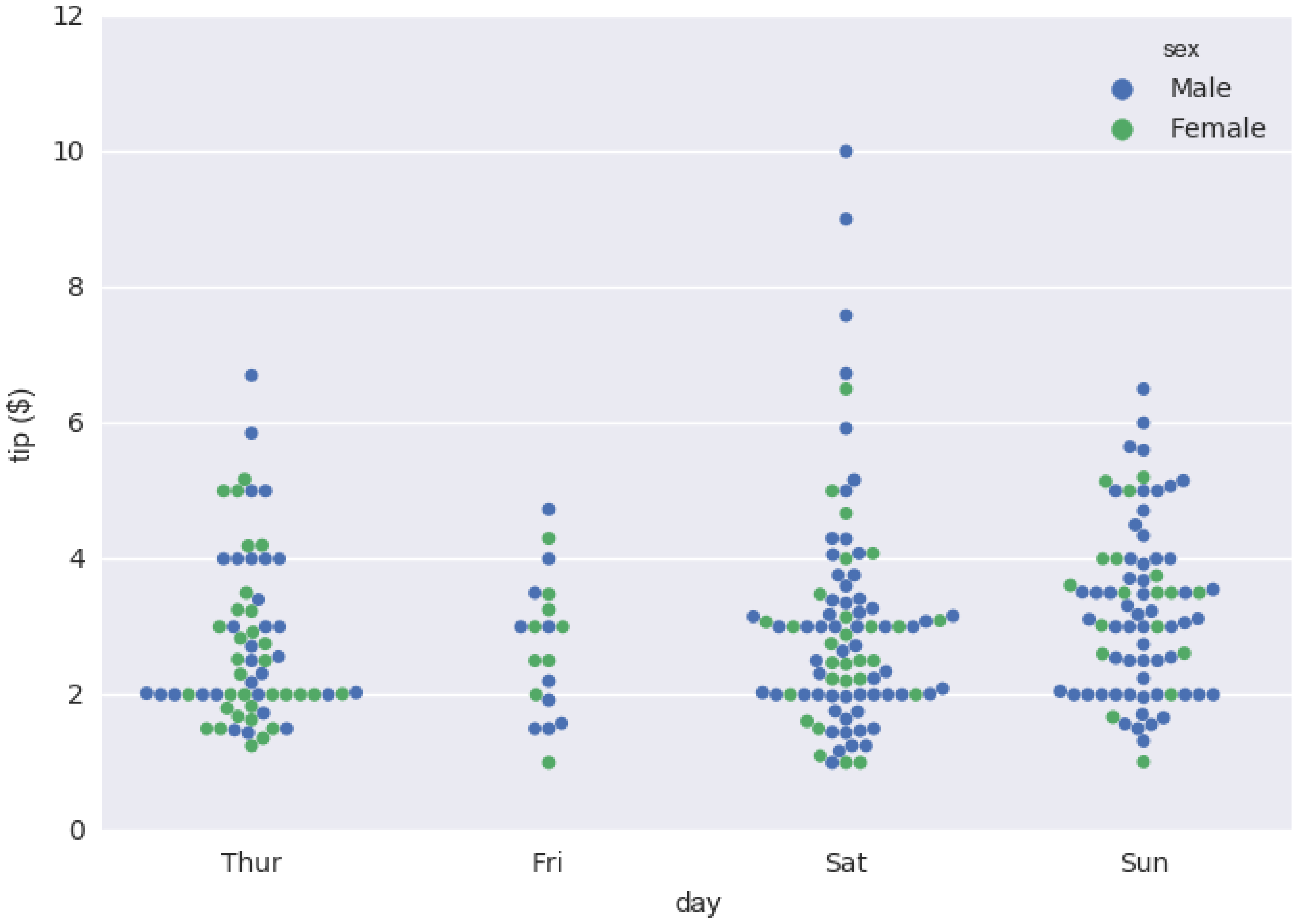


Using swarmplot()

```
sns.swarmplot(x='day',
              y='tip', data=tips)
plt.ylabel('tip ($)')
plt.show()
```

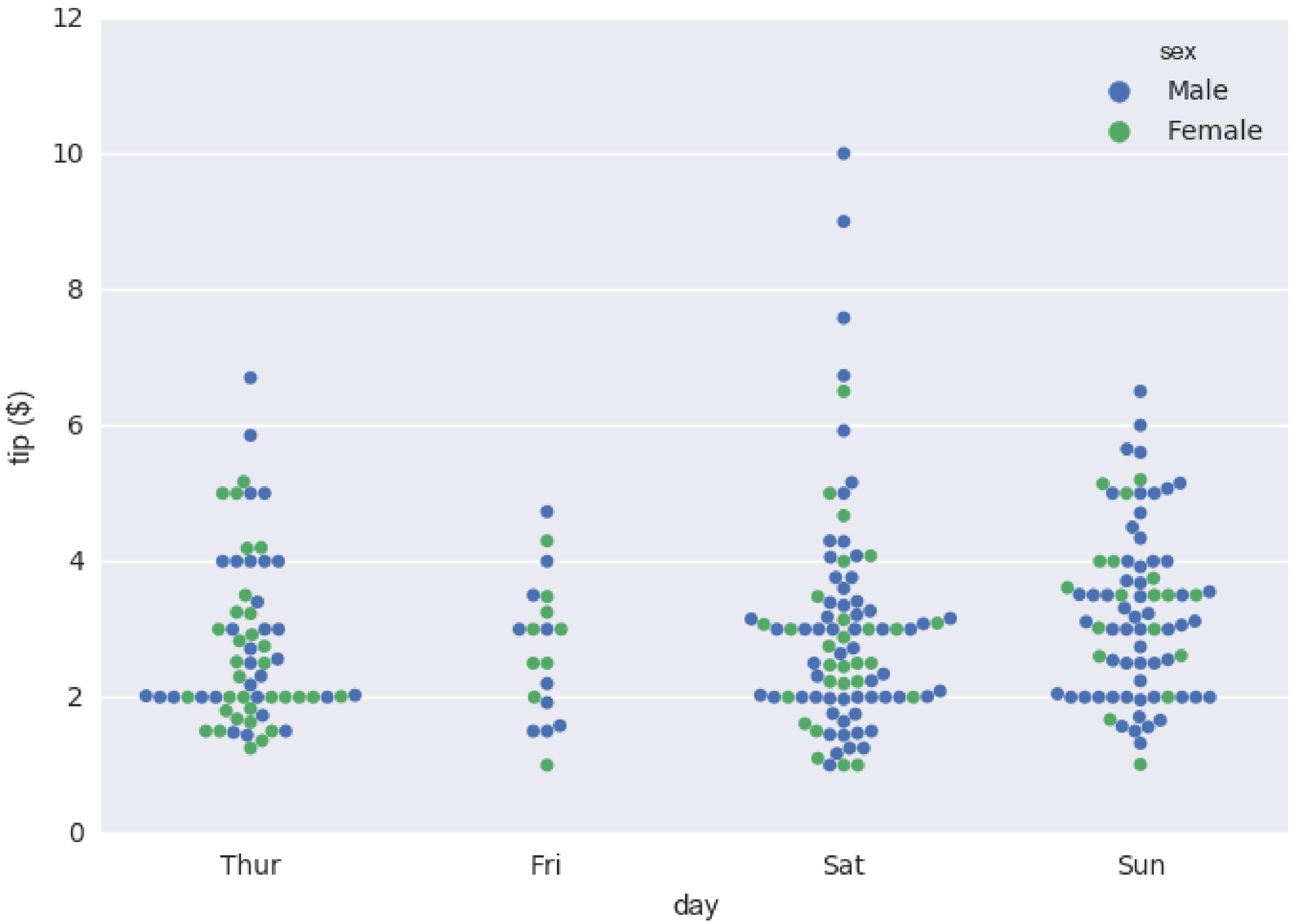


More grouping

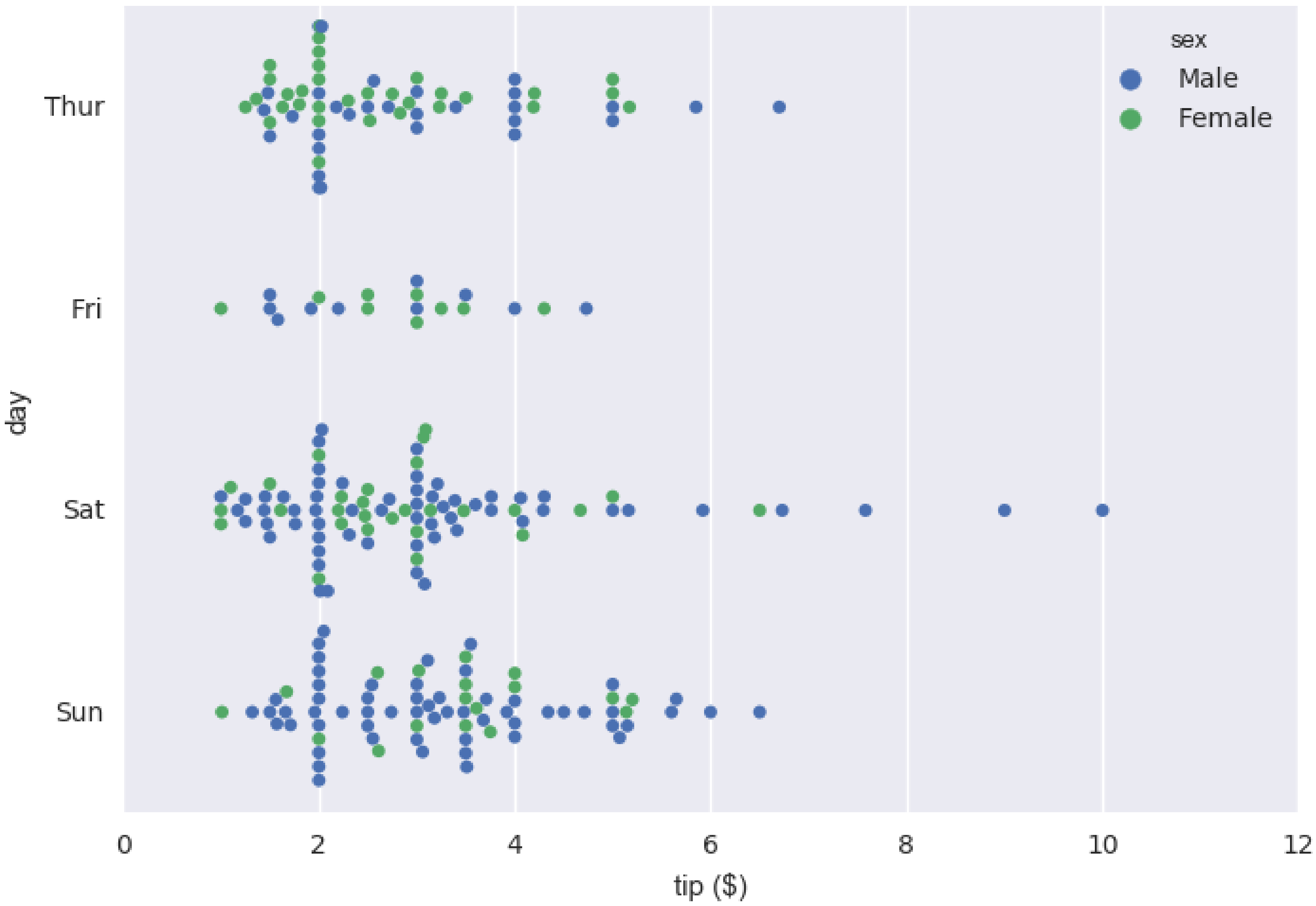


More grouping with swarmplot()

```
sns.swarmplot(x='day',
              y='tip', data=tips,
              hue='sex')
plt.ylabel('tip ($)')
plt.show()
```

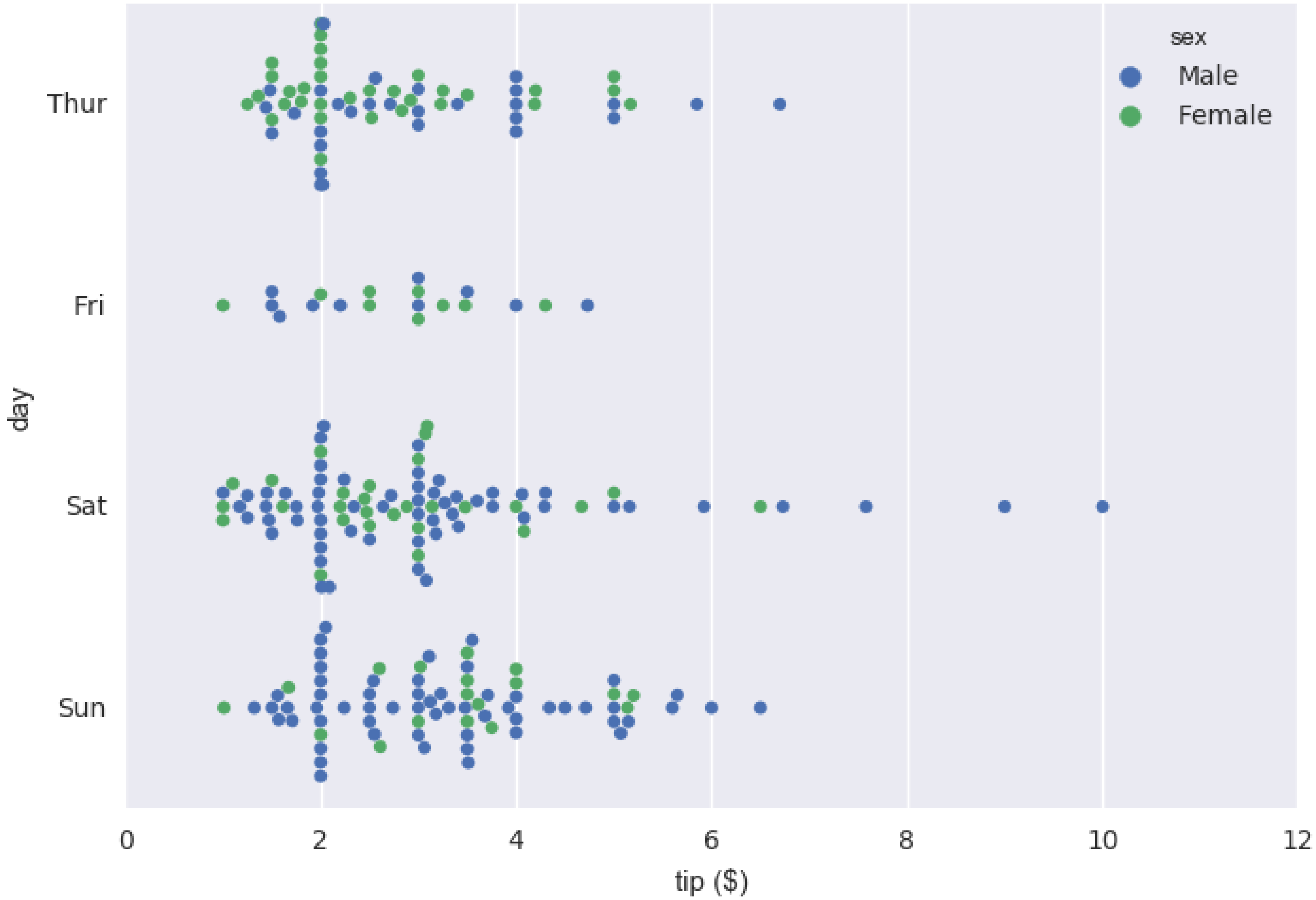


Changing orientation

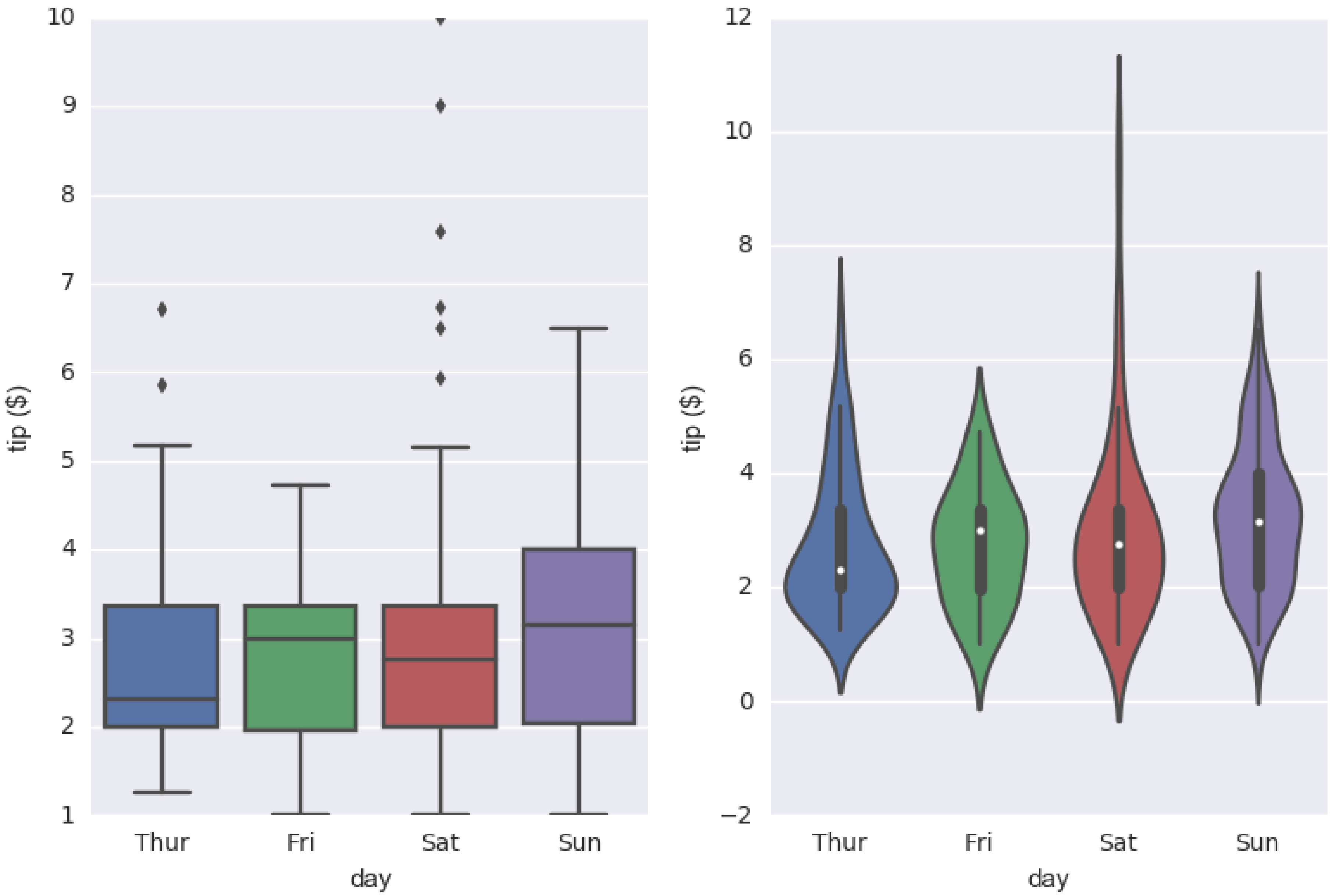


Changing orientation

```
sns.swarmplot(x='tip',
              y='day',
              data=tips,
              hue='sex',
              orient='h')
plt.xlabel('tip ($)')
plt.show()
```



Violin plot

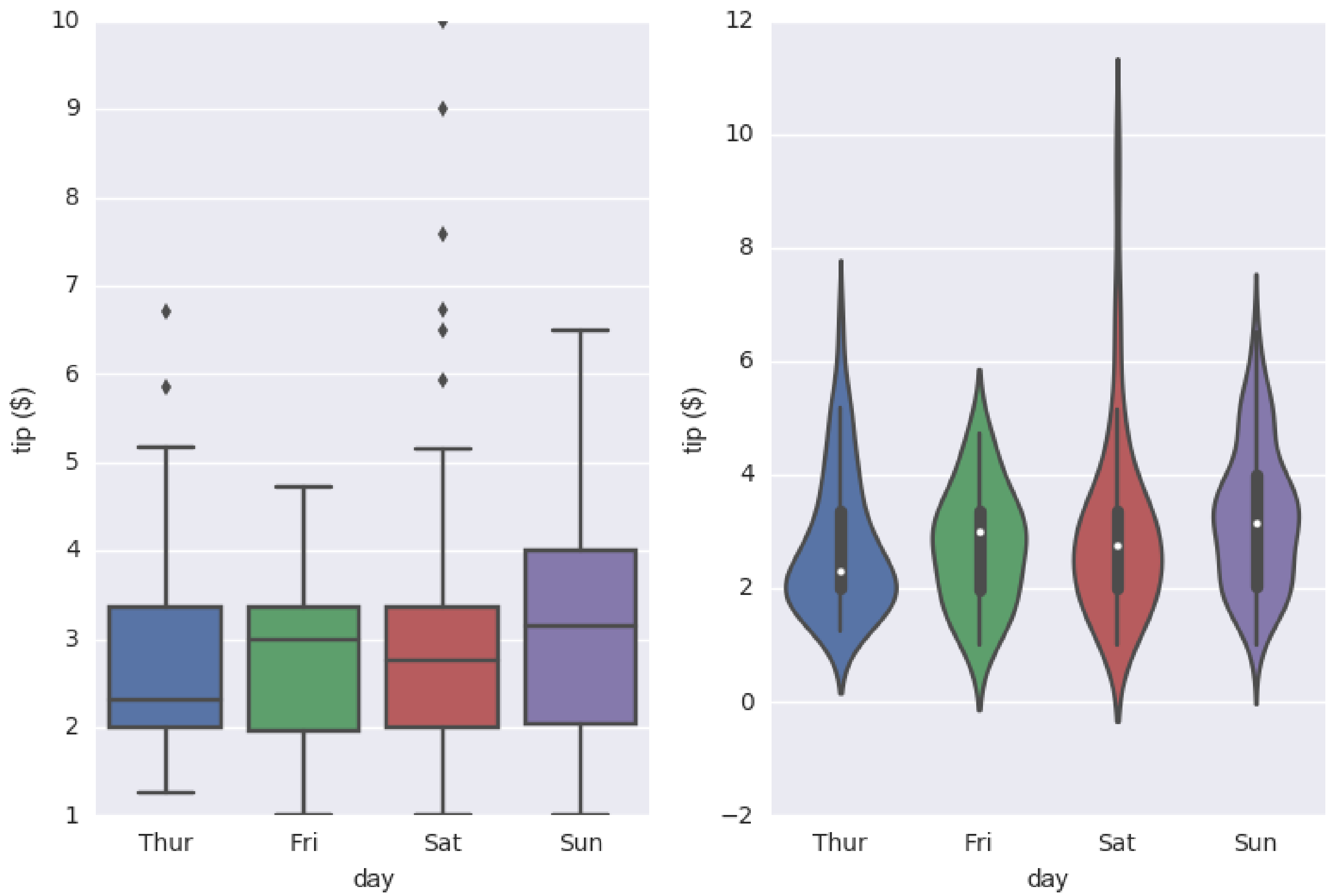

10
9
8
7
6
5
4
3
2
1
tip ($)
Thur
Fri
Sat
Sun
day
12
10
8
6
4
2
0
−2
tip ($)
Thur
Fri
Sat
Sun
day

Using violinplot()

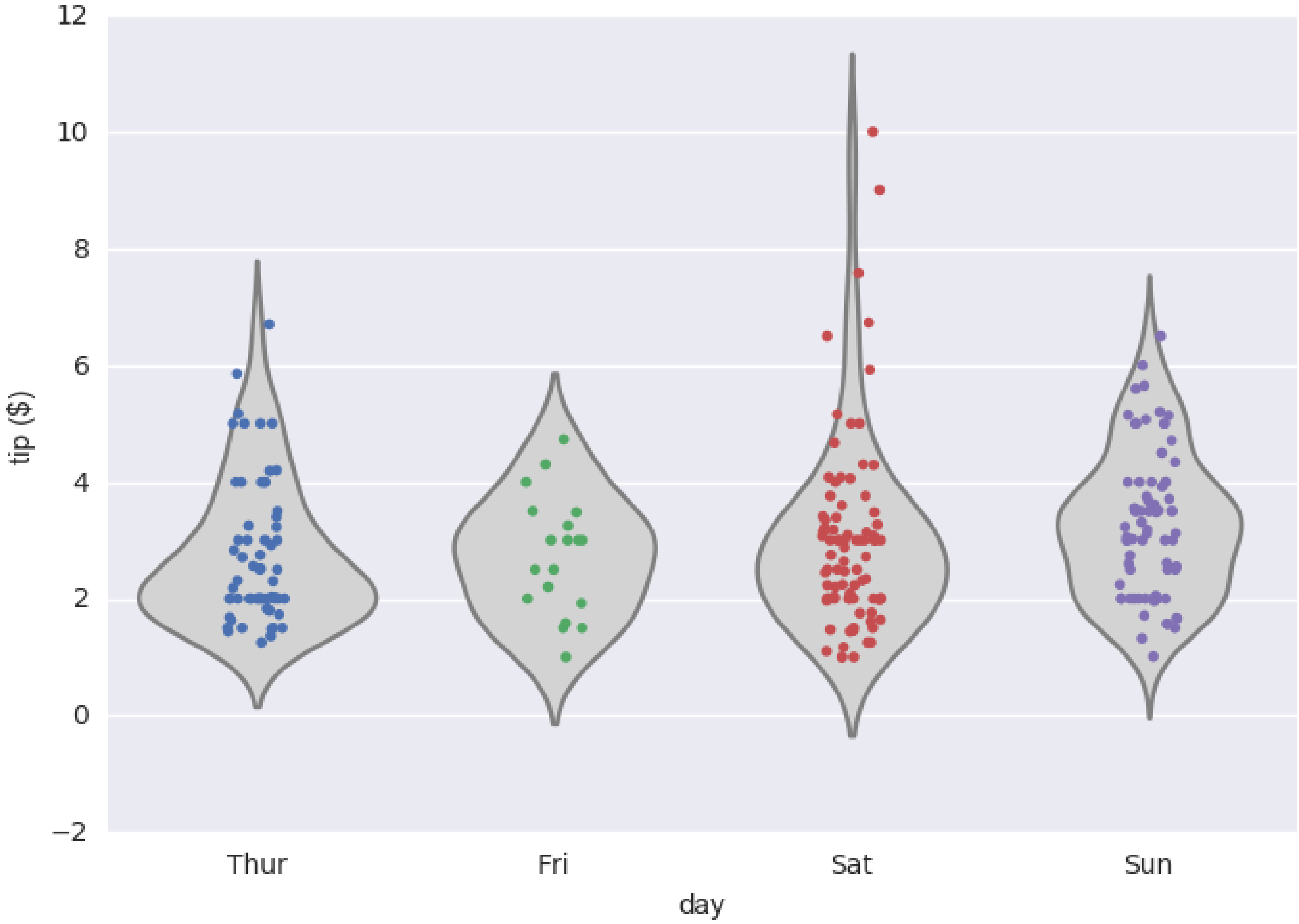
```
plt.subplot(1,2,1)
sns.boxplot(x='day',
            y='tip', data=tips)
plt.ylabel('tip ($)')

plt.subplot(1,2,2)
sns.violinplot(x='day',
               y='tip', data=tips)
plt.ylabel('tip ($)')

plt.tight_layout()
plt.show()
```



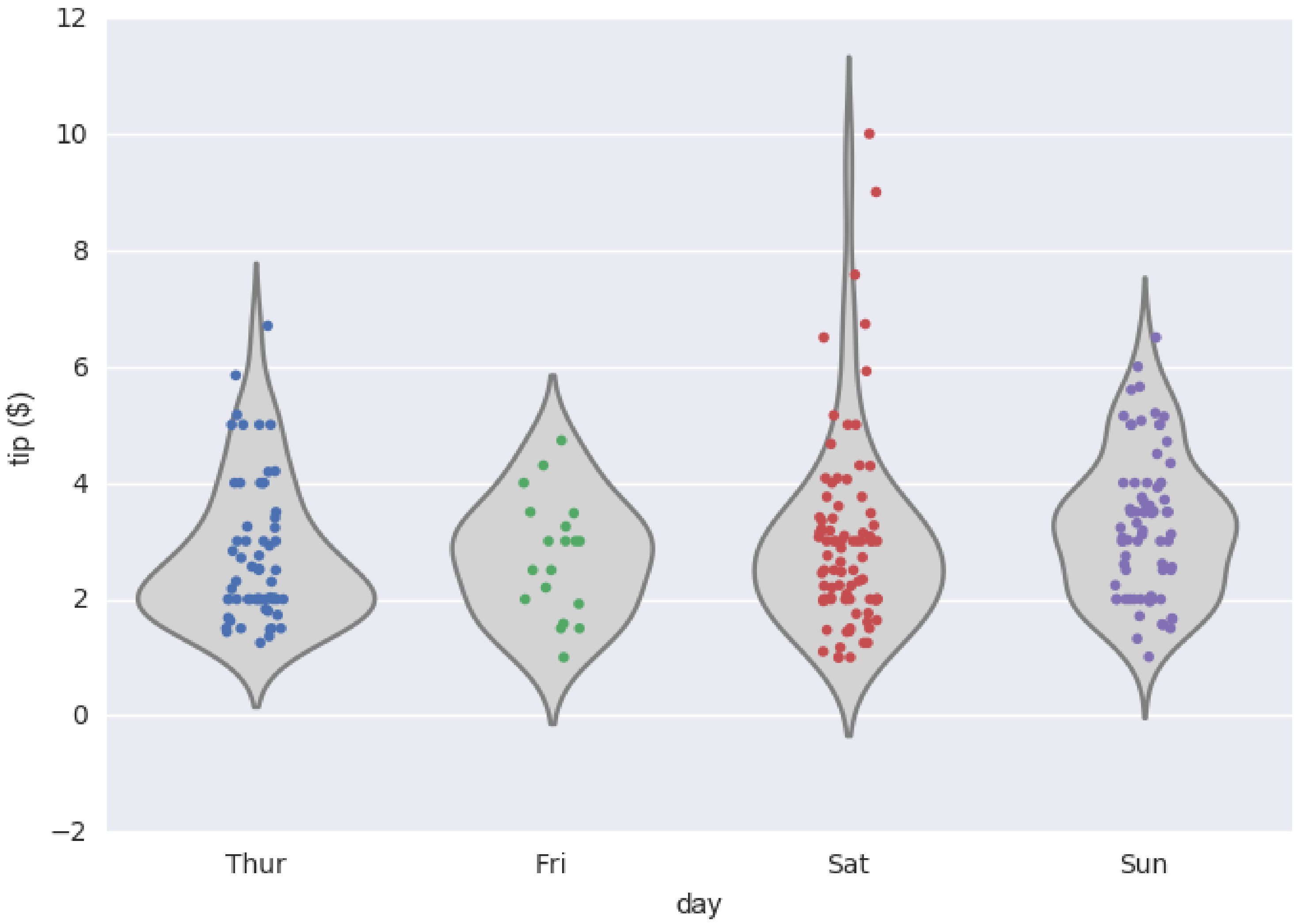
Combining plots



Combining plots

```
sns.violinplot(x='day',
               y='tip', data=tips,
               inner=None,
               color='lightgray')
sns.stripplot(x='day',
              y='tip', data=tips,
              size=4,
              jitter=True)

plt.ylabel('tip ($)')
plt.show()
```



Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Visualizing multivariate distributions

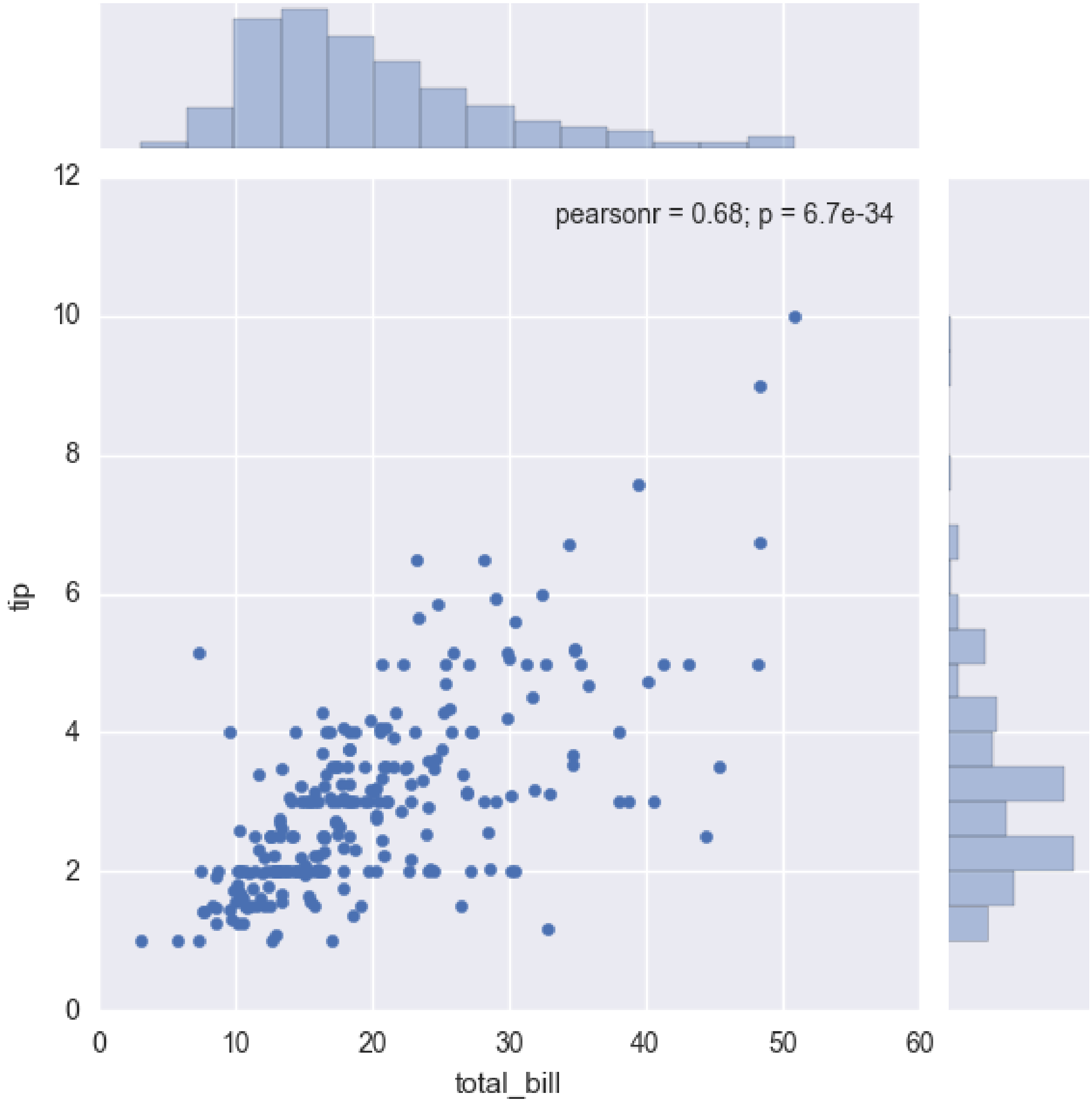
INTRODUCTION TO DATA VISUALIZATION IN PYTHON

Bryan Van de Ven
Core Developer of Bokeh

Visualizing data

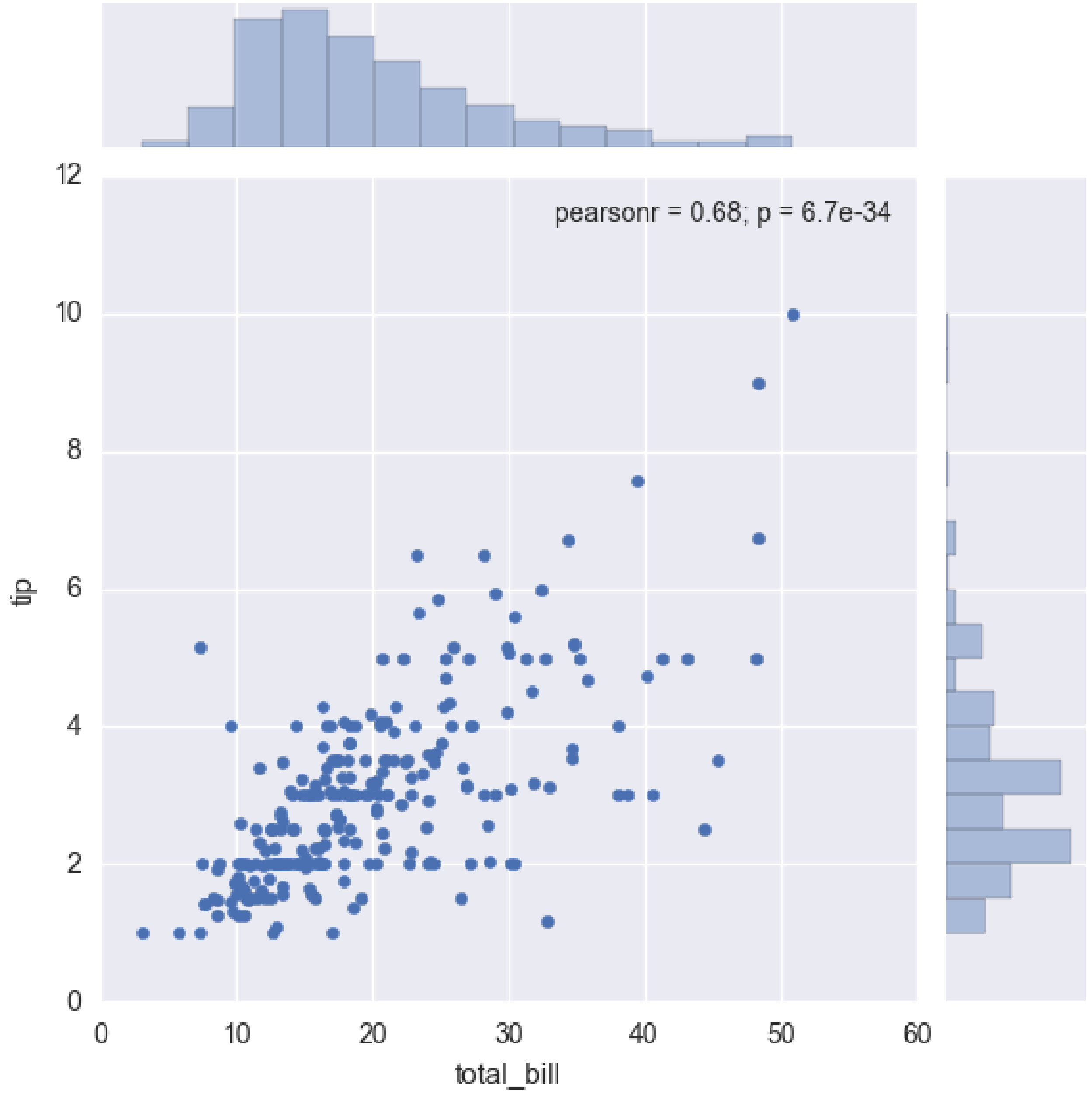
- Bivariate → "two variables"
- Multivariate → "multiple variables"
- Visualizing relationships in multivariate data
  - Joint plots
  - Pair plots
  - Heat maps

Joint plot

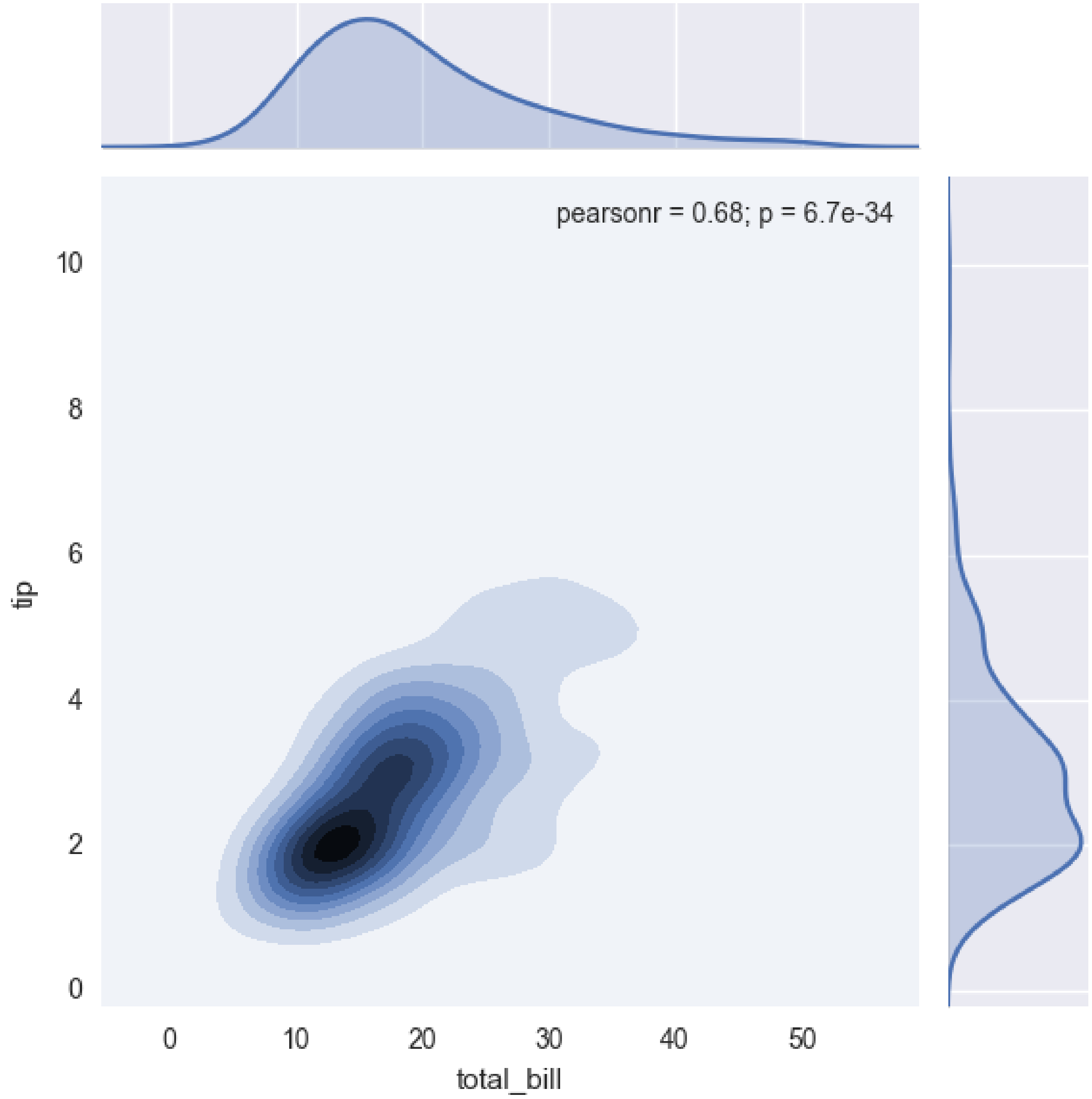

pearsonr = 0.68; p = 6.7e-34
12
10
8
6
4
2
0
tip
0
10
20
30
40
50
60
total_bill

Using jointplot()

```
sns.jointplot(x= 'total_bill',
              y= 'tip',
              data=tips)
plt.show()
```

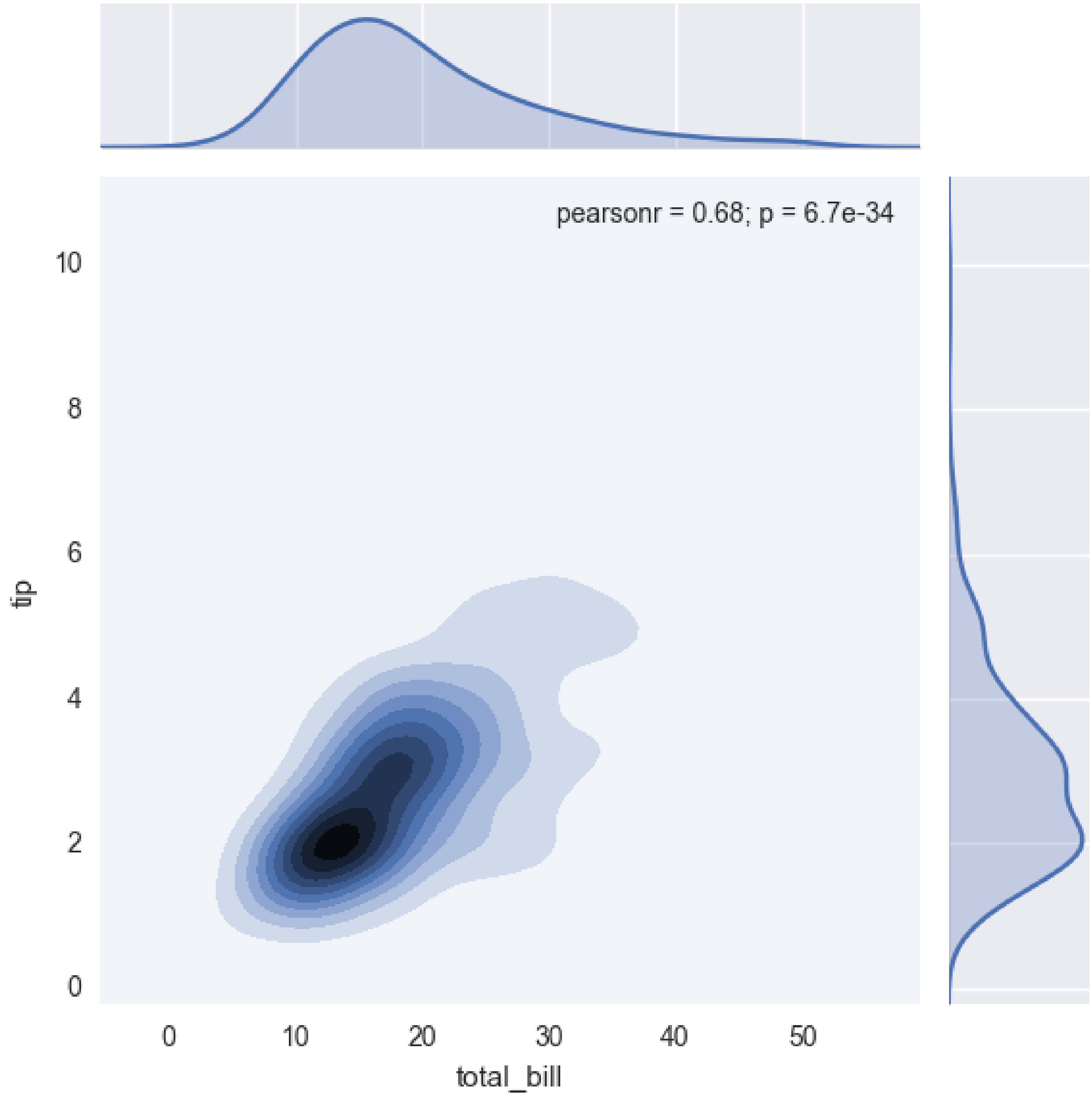


Joint plot using KDE

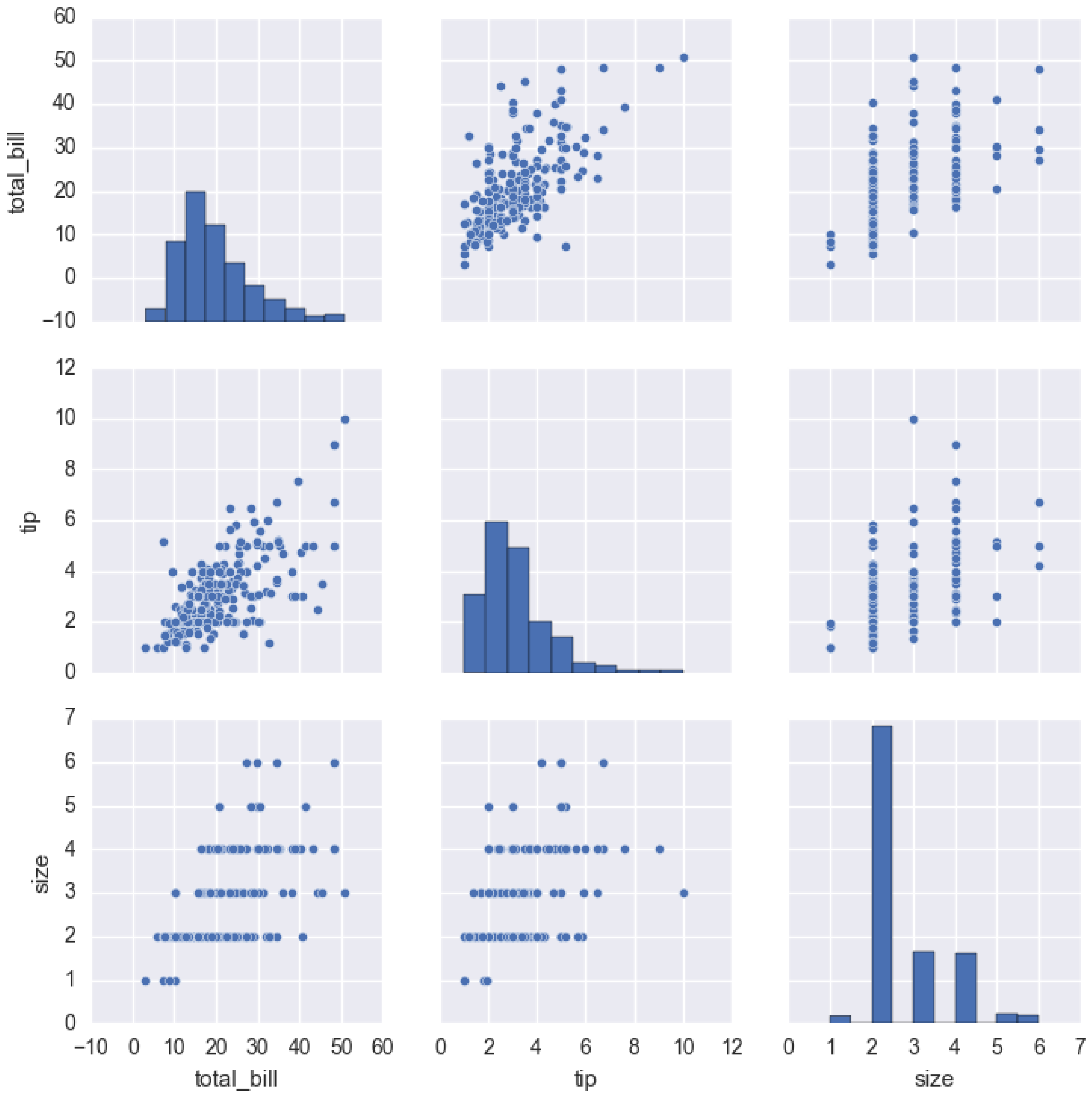


Using kde=True

```
sns.jointplot(x='total_bill',
              y= 'tip',
              data=tips,
              kind='kde')
plt.show()
```

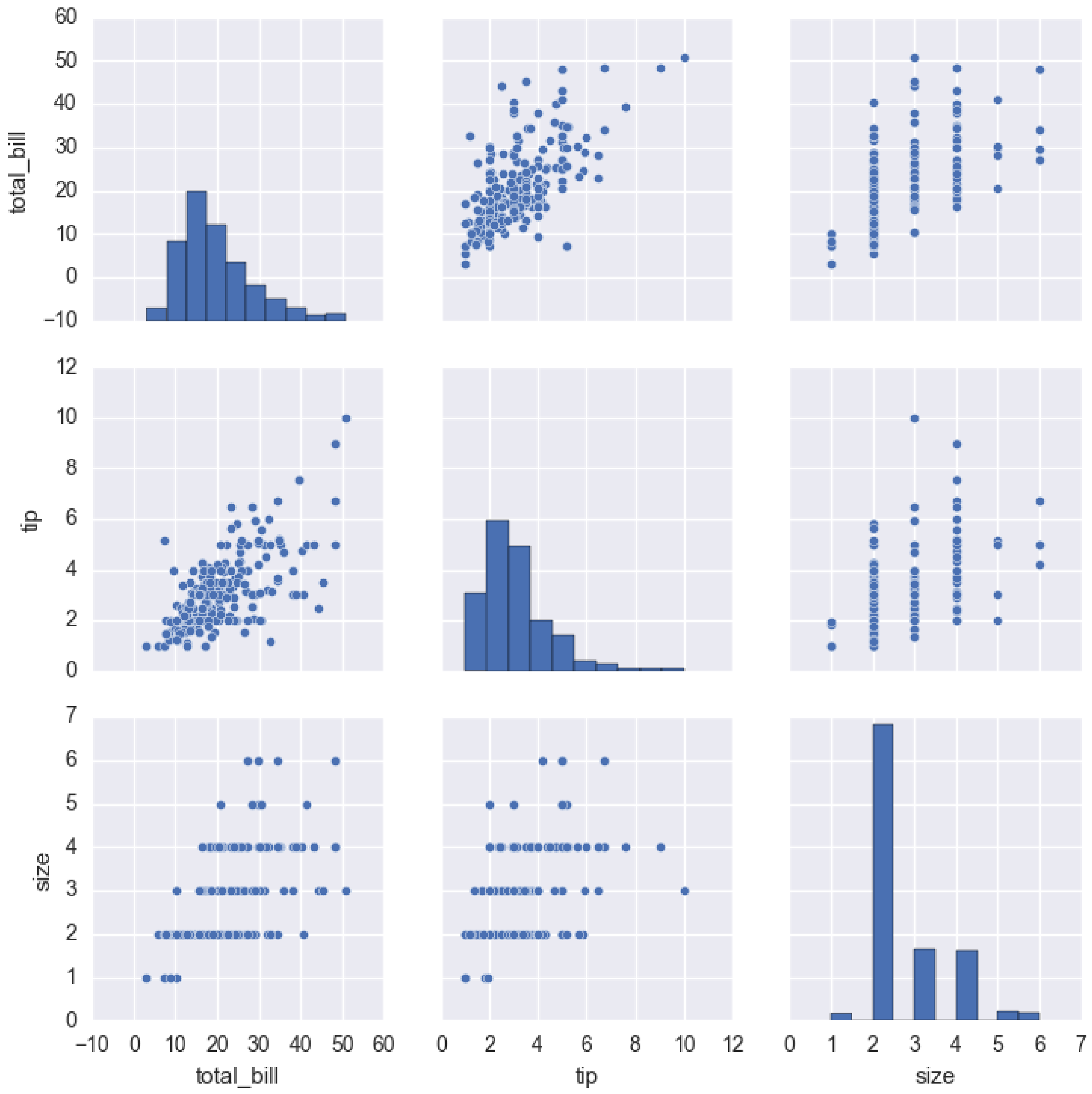


Pair plot


total_bill
tip
size
60
50
40
30
20
10
0
−10
12
10
8
6
4
2
0
7
6
5
4
3
2
1
0
−10
0
10
20
30
40
50
60
total_bill
0
2
4
6
8
10
12
tip
0
1
2
3
4
5
6
7
size

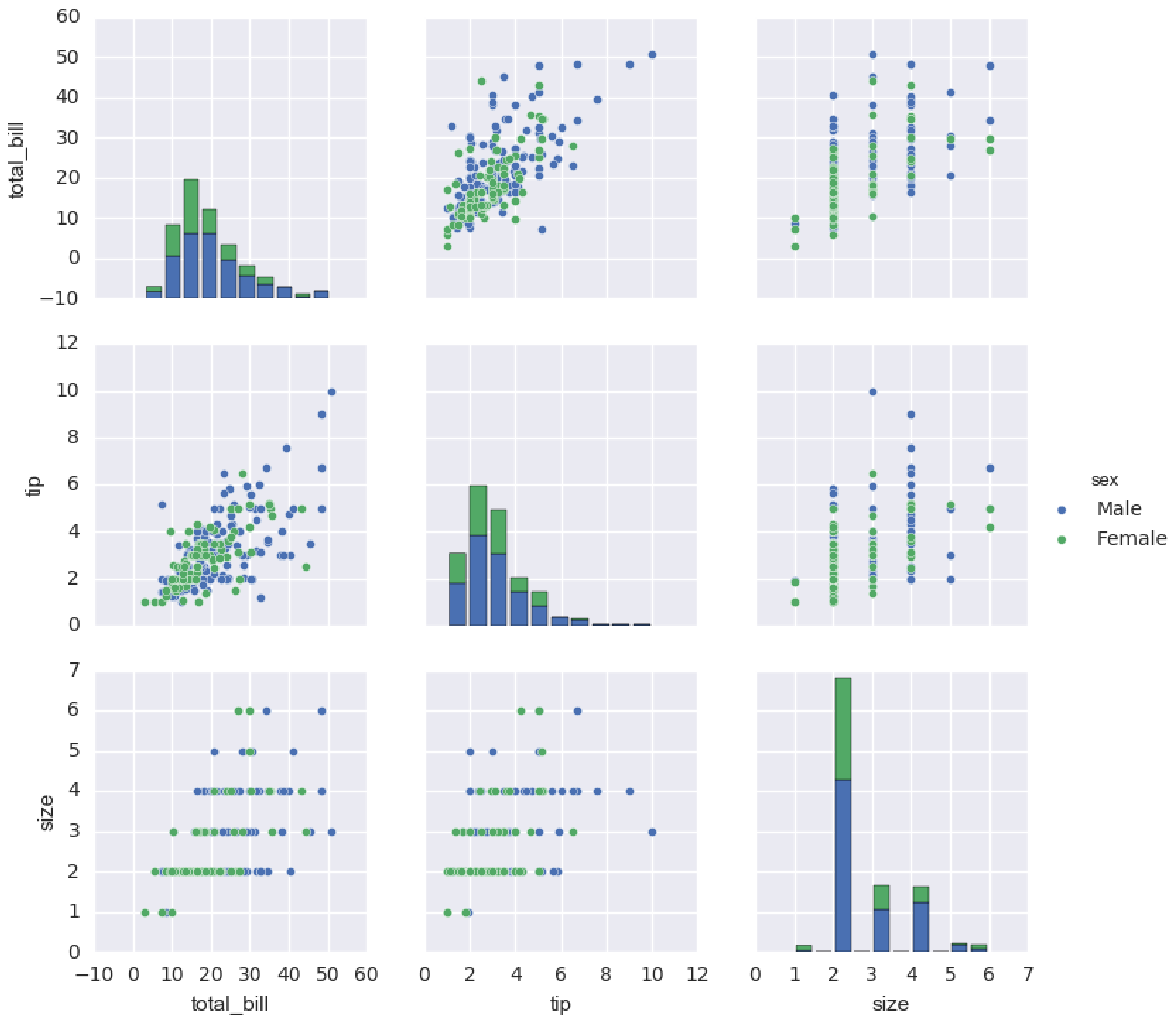
Using pairplot()

```
sns.pairplot(tips)
plt.show()
```

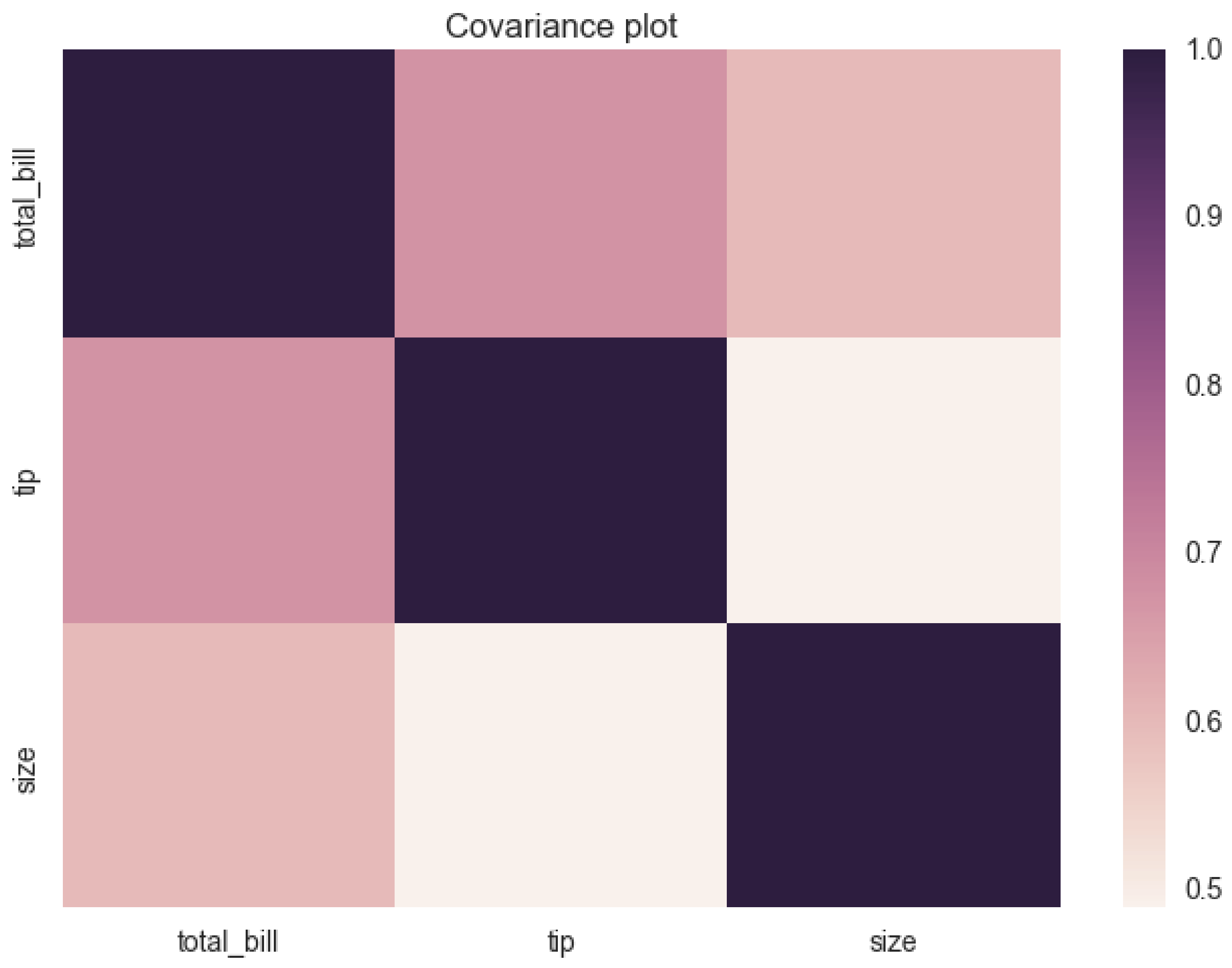


Using pairplot() with hue

```
sns.pairplot(tips,
             hue='sex')
plt.show()
```



Covariance heat map of tips data

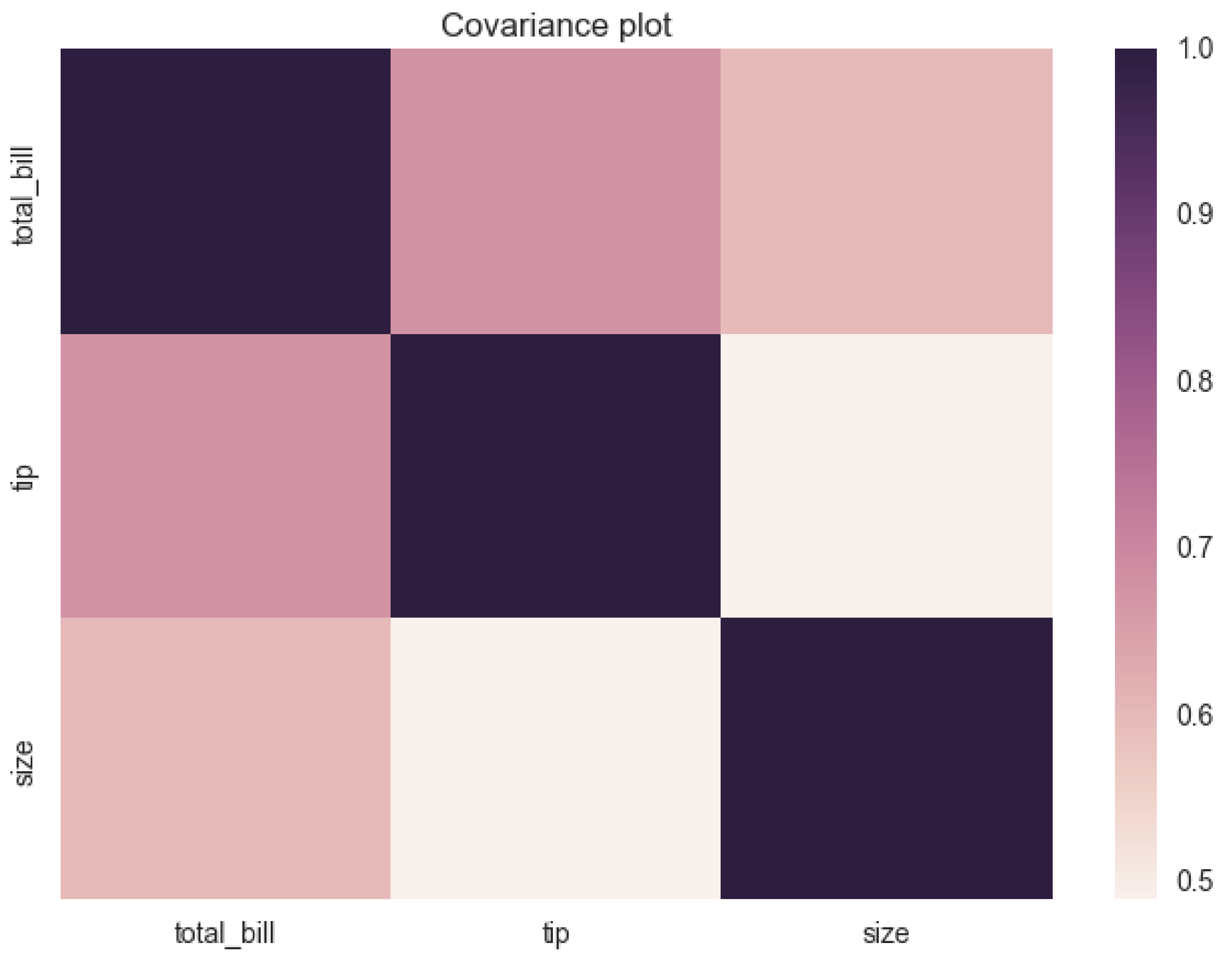


Using heatmap()

```
print(covariance)
```

```
            total_bill       tip      size
total_bill    1.000000  0.675734  0.598315
tip           0.675734  1.000000  0.489299
size          0.598315  0.489299  1.000000
```

```
sns.heatmap(covariance)
plt.title('Covariance plot')
plt.show()
```



Let's practice!

INTRODUCTION TO DATA VISUALIZATION IN PYTHON