Statistical estimation within categories

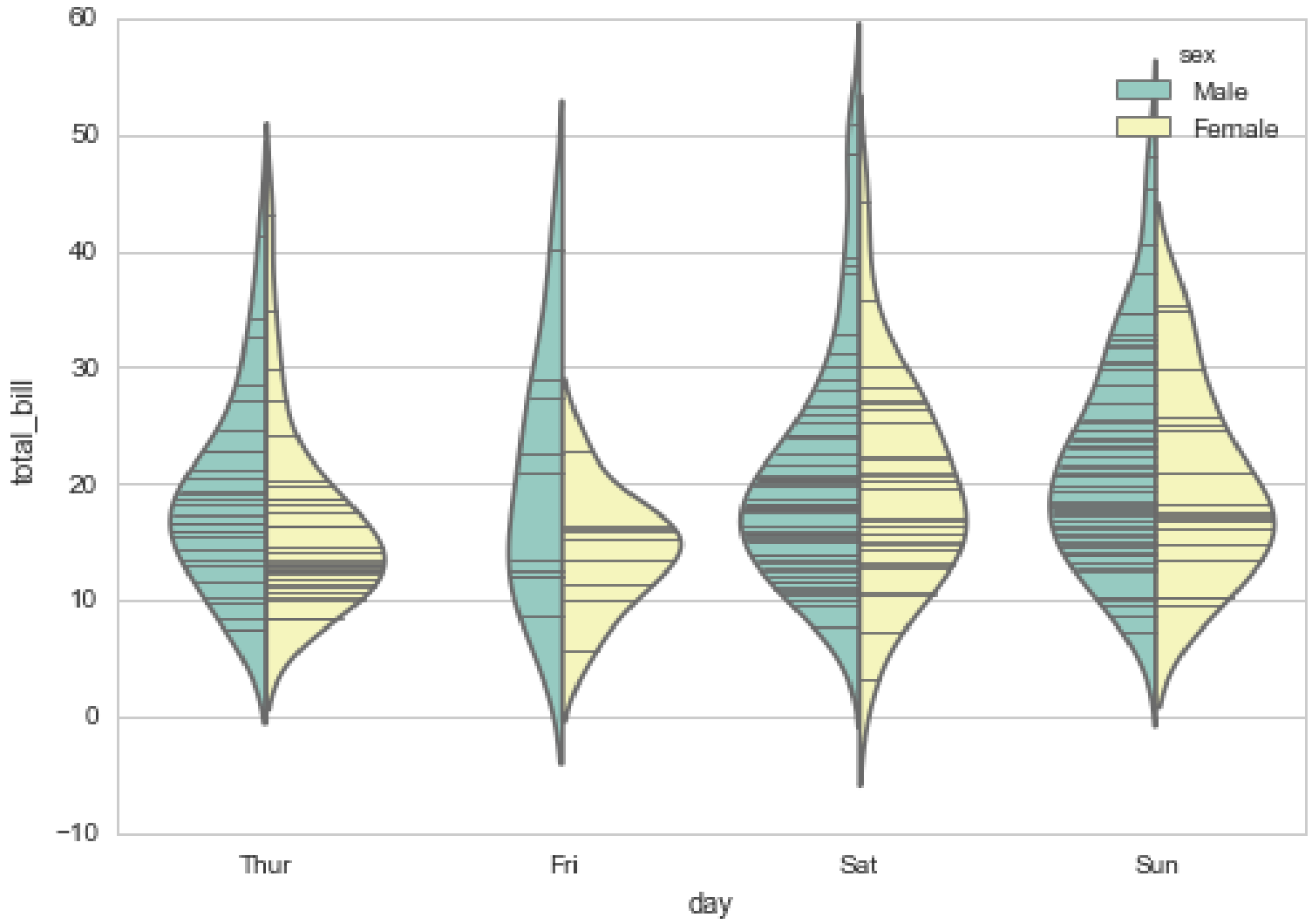
- Often, rather than showing the distribution within each category, you might want to show the central tendency of the values.
- Seaborn has two main ways to show this information, but importantly, the basic API for these functions is identical to that for the ones discussed above.

Bar plots

- A familiar style of plot that accomplishes this goal is a bar plot.
- In seaborn, the `barplot()` function operates on a full dataset and shows an arbitrary estimate, using the mean by default.
- When there are multiple observations in each category, it also uses bootstrapping to compute a confidence interval around the estimate and plots that using error bars:

Seaborn Workshop

```
sns.barplot(x="sex", y="survived",
      hue="class", data=titanic);
```

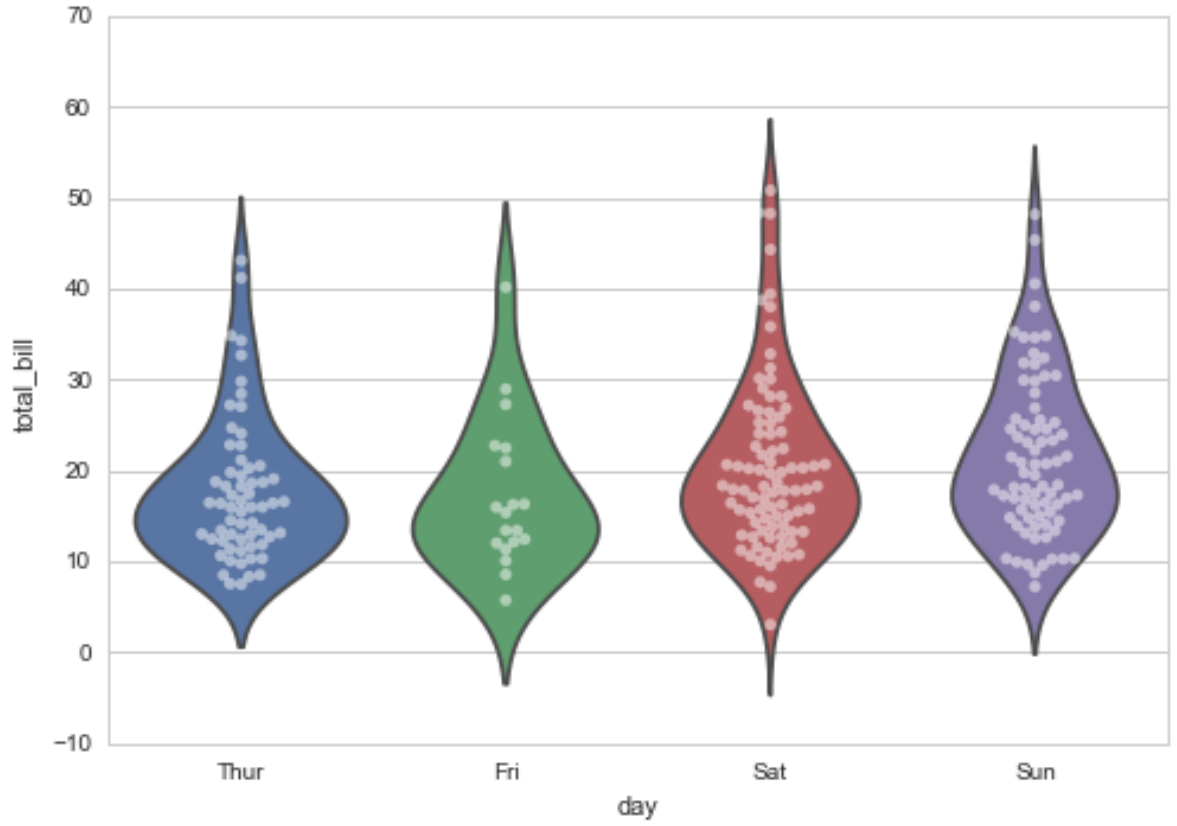


Seaborn Workshop

- A special case for the bar plot is when you want to show the number of observations in each category rather than computing a statistic for a second variable.
- This is similar to a histogram over a categorical, rather than quantitative, variable.
- In seaborn, its easy to do so with the `countplot()` function:

Seaborn Workshop

```
sns.countplot(x="deck", data=titanic, palette="Greens_d
```

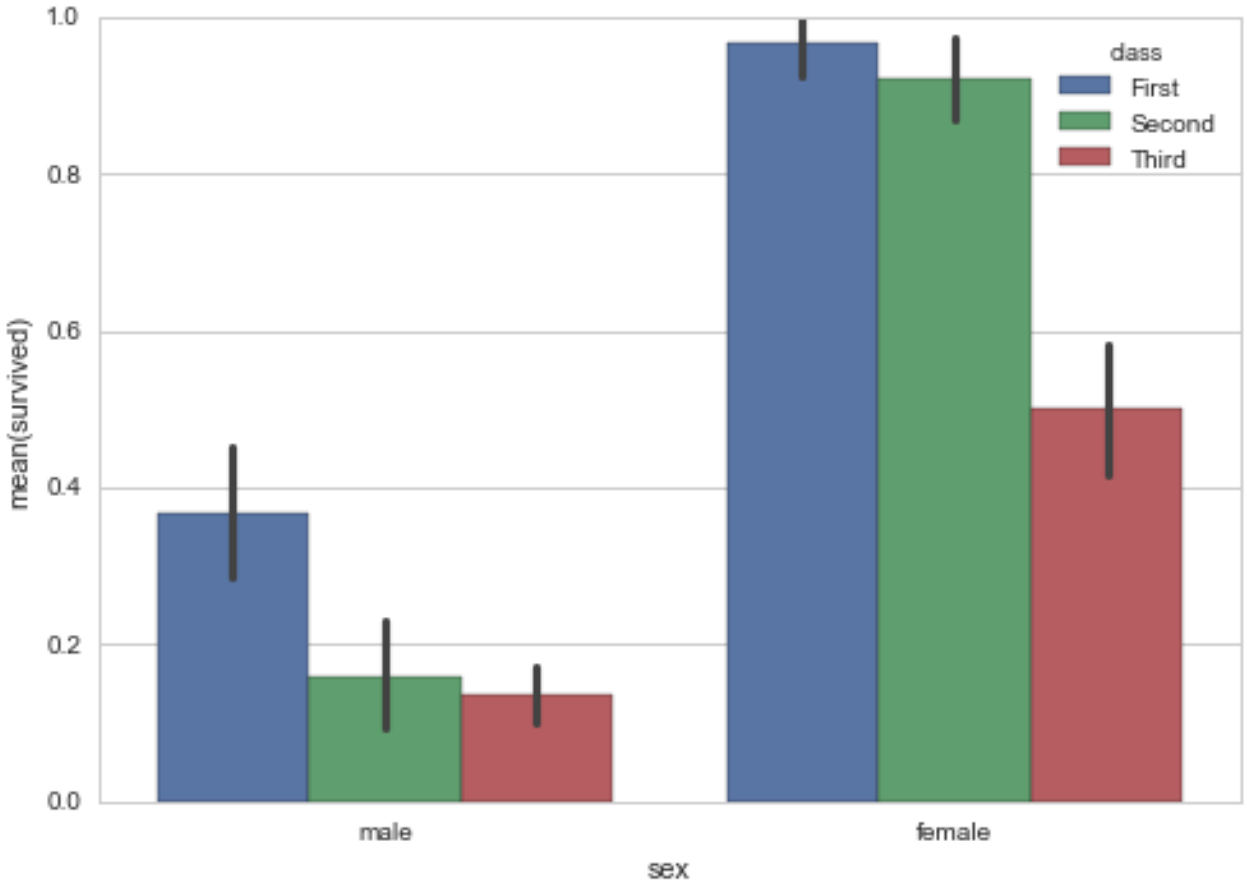


Seaborn Workshop

Both `barplot()` and `countplot()` can be invoked with all of the options discussed above, along with others that are demonstrated in the detailed documentation for each function:

Seaborn Workshop

```
sns.countplot(y="deck", hue="class",
      data=titanic, palette="Greens_d");
```

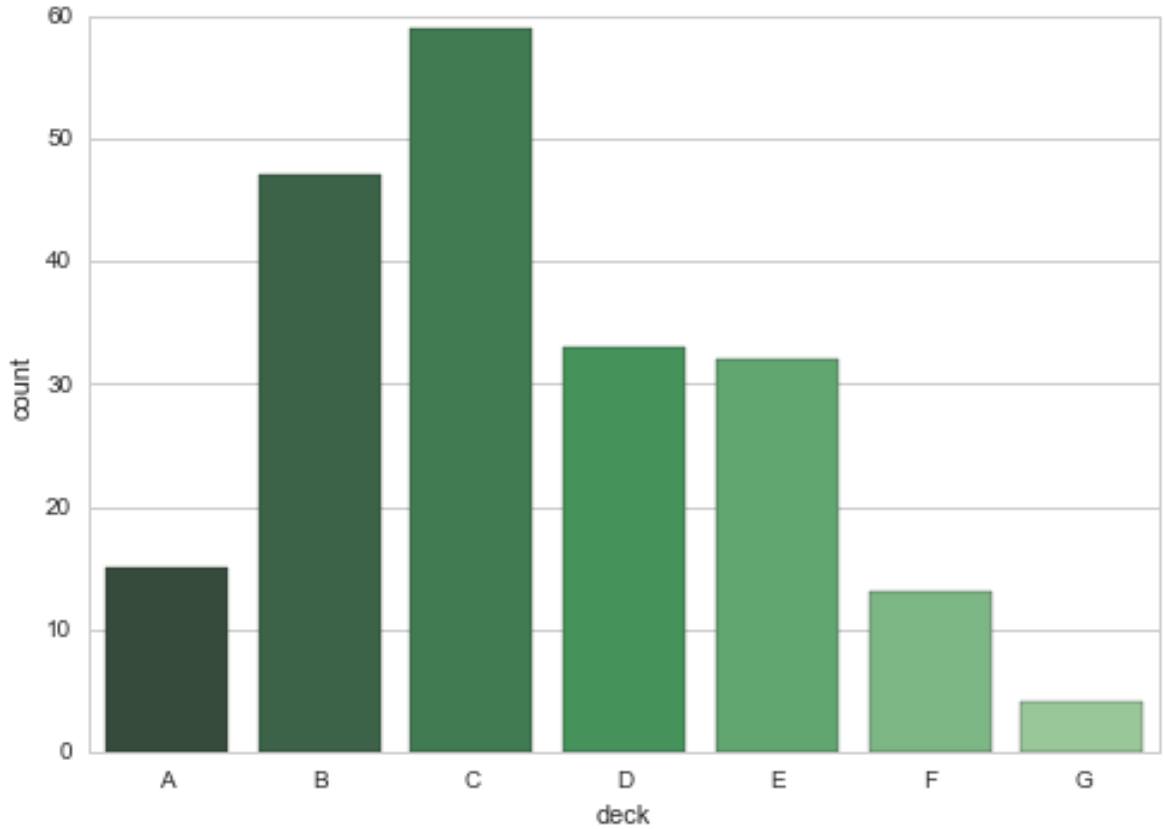


Point plots

- An alternative style for visualizing the same information is offered by the `pointplot()` function.
- This function also encodes the value of the estimate with height on the other axis, but rather than show a full bar it just plots the point estimate and confidence interval.
- Additionally, pointplot connects points from the same hue category.
- This makes it easy to see how the main relationship is changing as a function of a second variable, because your eyes are quite good at picking up on differences of slopes:

Seaborn Workshop

```
sns.pointplot(x="sex", y="survived", hue="class", data=
```



Seaborn Workshop

- To make figures that reproduce well in black and white, it can be good to use different markers and line styles for the levels of the hue category.

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

Seaborn Workshop