Course reader: *Descriptive statistics and visualizations*

- You need to know the difference between descriptive and inferential statistics:
  - *Descriptive statistics* are simply numbers that provide insights into data. For example, the mean (average) is a descriptive statistic for the central tendency, or expected value, of a set of numbers. There are lots and lots of descriptive statistics, and different statistics are useful for different kinds of data.
  - *Inferential statistics* provide information about how likely an effect is given the data that were collected, e.g., from an experiment.

- Here is a concrete example:
  - What is the average height of adult males in Belgium? (it's about 179 cm, in case you are curious!)
  - Are Dutch females significantly taller than German females?

  The first question is descriptive, because it is a number that describes a distribution. The second question is inferential, because it is asking whether an effect (height difference in a population) is a "real" effect or could be observed by chance. (I don't actually know the answer to the question, but I would guess that the answer is Yes.)

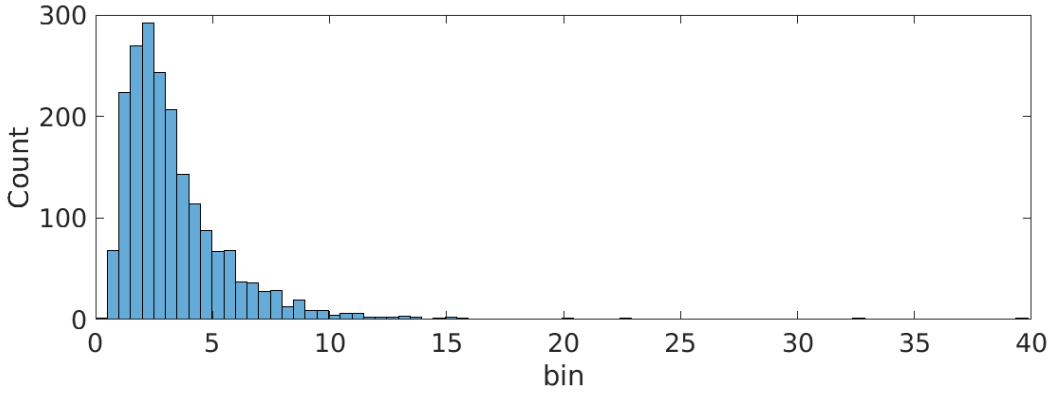
  This course focuses exclusively on descriptive, not inferential statistics.

- Mean and median are two measures of the central tendency of a distribution. The mean is the average, and you get it by summing all the numbers and dividing by the number of numbers. The median is the middle value and you get it by sorting the numbers and then finding the number that cuts the distribution into two equal parts (if there is an even number of numbers, then the median is the average of the two middle values).

- Comparing the mean and median is insightful:
  - If they are roughly the same, the distribution is close to normal (Gaussian).
  - If the mean is higher, the distribution is skewed to the right.
  - If the mean is lower, the distribution is skewed to the left.

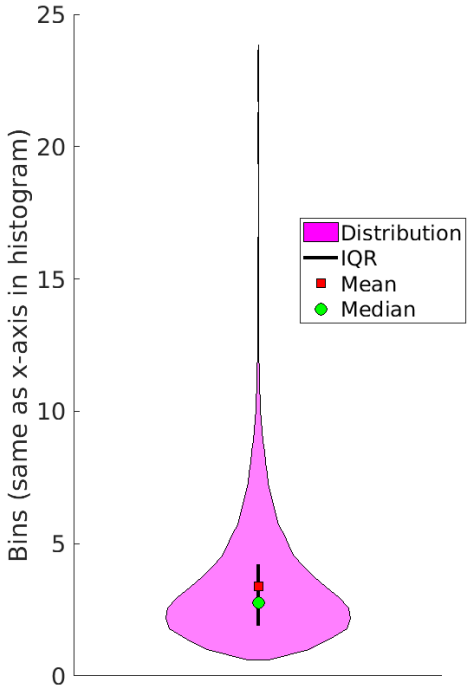
- Interquartile range is the range of values that contains the middle 50% of the data distribution. Think of it this way: The median divides the data into two equal-sized halves. Now compute the median of each half. Those 3 medians (the real median and the two half-medians) split the data into 4 equal-sized bins. Then take the distance between the 1^{st} and 4^{th} quartiles. Easy peasy!

- We've gone from median (divide the data into halves) to interquartile range (divide the data into quarters)... why stop there? What about creating, say, 40 bins that range from smallest to largest value? Then you could count the number of data values in each bin, and show the result as a plot. The plot would have bin value on the x-axis and count (or sometimes converted to probability) on the y-axis. This is the idea of a **histogram**. Below is an example of a histogram.



  The number of bins to use is the key parameter of the histogram, and there is no clear answer. A few guidelines are presented in the exercises, but mostly you just have to pick an appropriate value based on the data.

- Now imagine rotating the histogram so it stands up, then smooth it a bit and reflect it. Then you have a **violin plot**! Below is the same data as above but in violin-form.



- Violin plots are beautiful representations of the data, and have a major advantage that they can show multiple distributions in a way that histograms don't really convey.

Exercises

1. Of the following descriptive statistics, which will change if the ordering of data points were to change (e.g., if the order of the data values were randomly shuffled)? Mean, median, standard deviation, interquartile range.

2. Two of the common "rules" for the number of bins in a histogram (they're called "rules" but really they are loose suggestions) are as follows (in these equations, b is the number of bins, n is the number of data points in the distribution, x is the vector of data values, and q is the interquartile range):

- *Sturges' suggestion*: $b = 1 + \lceil \log_2 n \rceil$ ($\lceil \rceil$ means to round up to the nearest integer).
- *Freedman-Diaconis' suggestion*: $b = \lceil \frac{\max(x) - \min(x)}{2qn^{-1/3}} \rceil$.

Implement these two suggestions, and $b = 50$ bins, for the following data, and form an opinion of the three options.

```
n = 10000;  rng(1); % ensure reproducibility
data = [exp(.6*randn(1,n)+1)+2 randn(1,n)];
```

3. You might wonder why I prefer writing out code when I code just download some function from the Internet. There are several good answers to this, but here're two: (1) When you write your own code, you *really* understand what's happening to the data; (2) when you write your own code, you can modify it in ways that the downloaded code might not be able to do.
This exercise is about the latter. Modify the violin plot code to make the left and right side of the violin show distributions from different datasets.

Answers

1. None! These descriptive statistics are all based purely on data distributions, not temporal ordering. However, there are other descriptive statistics, such power spectra or autocorrelation spectra, that are highly sensitive to temporal order.

2. Below is the code. Sturges' suggestion generally produces poor results for large N. In this case, the Freedman-Diaconis suggestion gives 63 bins, which looks good. Freedman-Diaconis often gives good recommendations, because it is based on the *information* in the data (IQR) not only the number of data points. But don't blindly trust it without testing other values (try, e.g., N=100).

PYTHON

```
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt

fig,ax = plt.subplots(1,3,figsize=(10,3))

# create data
n = 1000
data = np.random.randn(n)

# Sturges' rule
r = int( np.ceil(np.log2(n))+1 )
ax[0].hist(data,r)
ax[0].set_title('Sturges: %s' %r)

# Friedman-Diaconis rule
r = 2*stats.iqr(data)*n**(-1/3)
b = int( np.ceil( (max(data)-min(data) )/r ) )
ax[1].hist(data,b)
ax[1].set_title('F-D: %s' %b)

# 50 bins
b = 50
ax[2].hist(data,b)
ax[2].set_title('%s bins' %b)

plt.show()
```

MATLAB

```
% create data
n = 1000;
data = randn(n,1);
```

```
% Sturges rule
subplot(131)
r = ceil(log2(n))+1;
hist(data,r), title([ 'Sturges: ' num2str(r) ])

subplot(132)
r = 2*iqr(data)*n^(-1/3);
b = ceil( (max(data)-min(data) )/r );
hist(data,b), title([ 'F-D: ' num2str(b) ])

subplot(133)
hist(data,50), title('b=50')
```

3. This one is less difficult then you might initially think. You need to create two datasets and then two histograms (I called them `x1,y1` and `x2,y2`). The mean, median, and IQR would need to be shifted off-center, or you could leave them out of the plot. The body of the violin is then:

PYTHON

```
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
from matplotlib.patches import Polygon

# generate data
n = 10000
data1 = np.random.randn(n)
data2 = np.random.randn(n)+.5

# create histograms
y1,x1 = np.histogram(data1,50)
y2,x2 = np.histogram(data2,50)

x1 = x1[0:-1]
x2 = x2[0:-1]

y = np.concatenate((y1,-y2[::-1]))
x = np.concatenate((x1, x2[::-1]))

# draw patch
y = np.vstack((y,x)).T
p = Polygon(y,facecolor='m',alpha=.3)

fig, ax = plt.subplots()
ax.add_patch(p)
```

```
ax.set_ylim([-4,4])
ax.set_xlim([-1000,1000])
plt.show()
```

MATLAB

```
% generate data
n = 10000;
data1 = randn(n,1);
data2 = randn(n,1)+.5;

% create histograms
[y1,x1] = hist(data1,50);
[y2,x2] = hist(data2,50);

% draw patch
patch([y1 -y2(end:-1:1)],[x1 x2(end:-1:1)],'m','facealpha',.5)
```