Chapter 3

The Pandas essentials for data visualization

Objectives (part 1)

Applied

1. Use the Pandas plot() method to create these types of plots:

   line plot
   area plot
   scatter plot
   bar plot
   histogram
   density plot
   box plot
   pie plot

2. Use the parameters of the Pandas plot() method to enhance a plot in these ways:

   add a title, x and y labels, and grid lines

   rotate the tick labels

   set the x- and y-axis limits

Objectives (part 2)

3. Use the parameters of the Pandas plot() method to create a plot that has subplots.
4. Chain the Pandas plot() method to methods that prepare the data for the plot() method.

Knowledge

1. List three data visualization libraries for Python.
2. Distinguish between long data and wide data and describe their effects on the Pandas plot() method.
3. Describe the way the Pandas plot() method works when no parameters are coded.

Objectives (part 3)

4. Describe these types of plots:

   line plot
   scatter plot
   bar plot
   histogram
   box plot

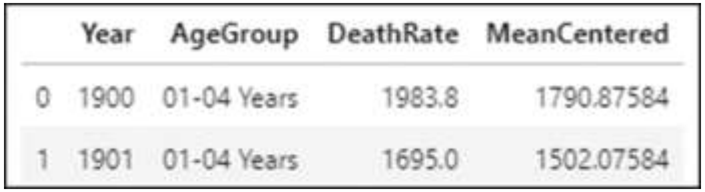
Data visualization libraries for Python

```
matplotlib
pandas
seaborn
altair
ggplot
```

Data visualization can help you…

- Understand your data more easily.
- See the relationships between variables.
- Spot unusual datapoints like outliers.

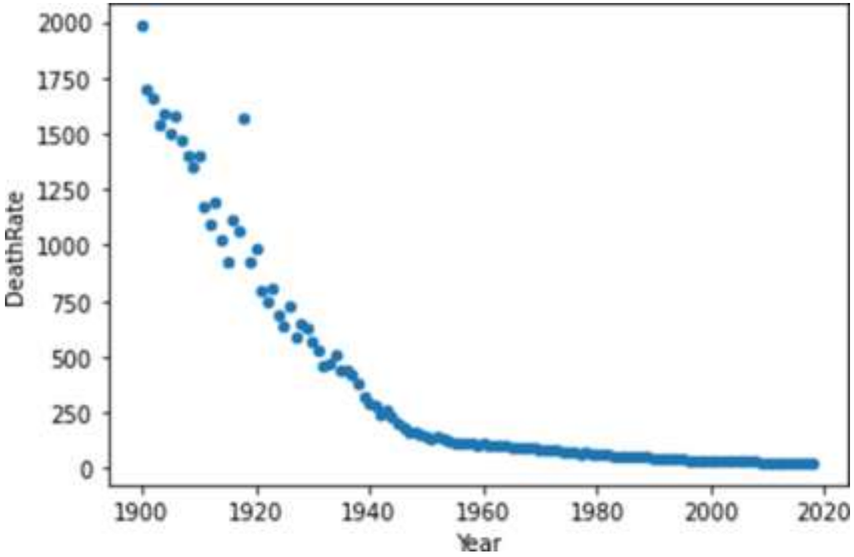
The mortality data in long form (mortality_data)



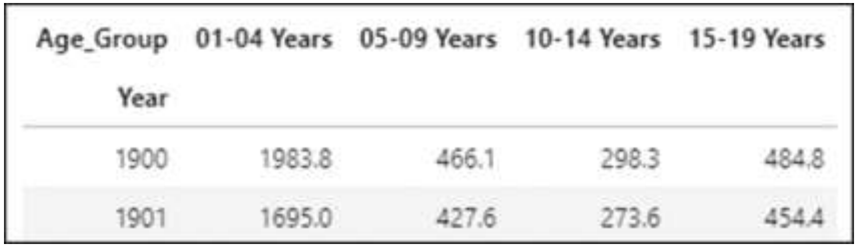
	Year	AgeGroup	DeathRate	MeanCentered
0	1900	01-04 Years	1983.8	1790.87584
1	1901	01-04 Years	1695.0	1502.07584

A scatter plot derived from the long data

```
mortality_data.query('AgeGroup == "01-04 Years"') \
    .plot.scatter(x='Year', y='DeathRate')
```



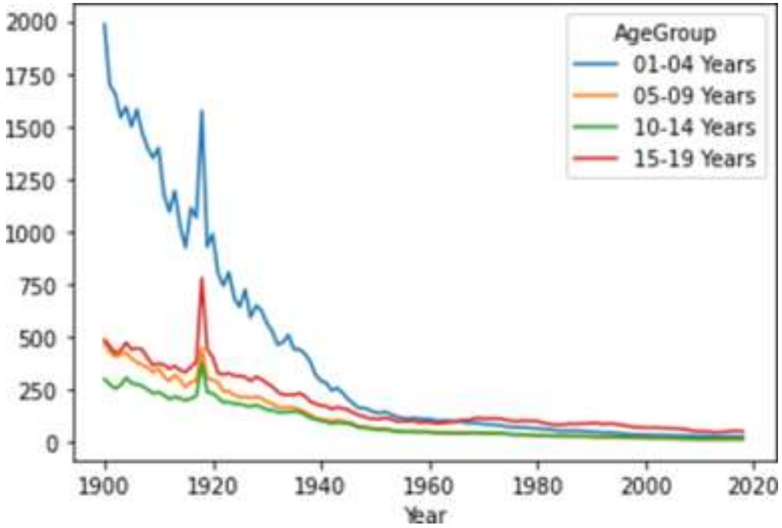
The mortality data in wide form with Year as the index (mortality_wide)



Age_Group Year	01-04 Years	05-09 Years	10-14 Years	15-19 Years
1900	1983.8	466.1	298.3	484.8
1901	1695.0	427.6	273.6	454.4

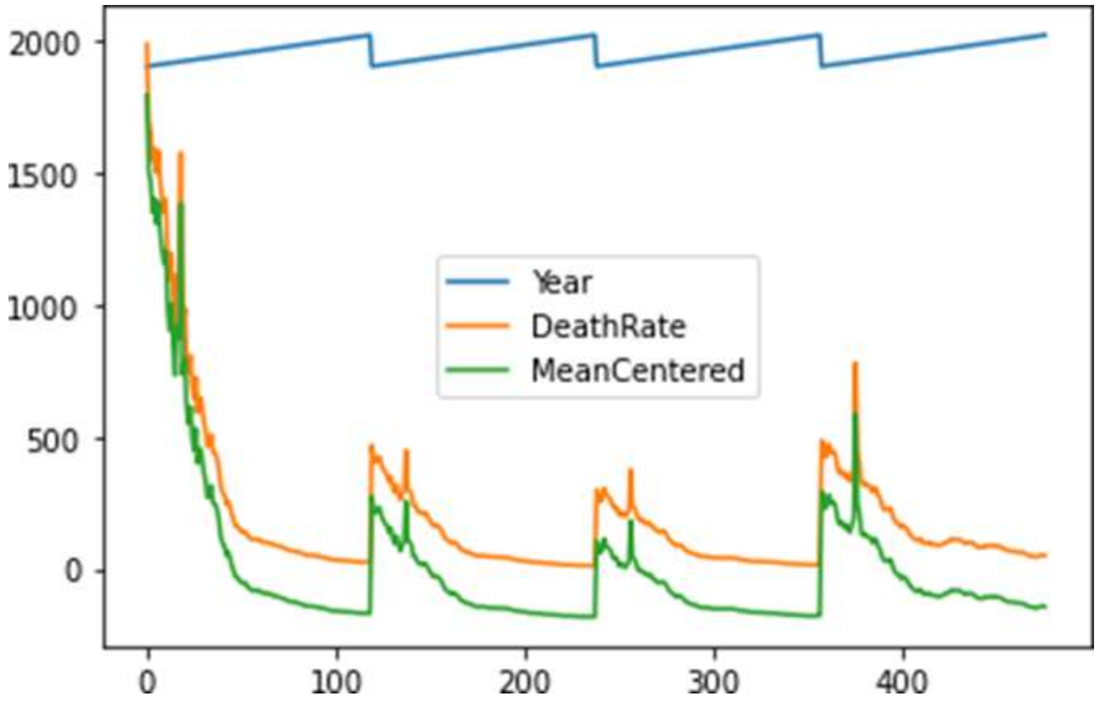
A line plot derived from the wide data

```
mortality_wide.plot()
```



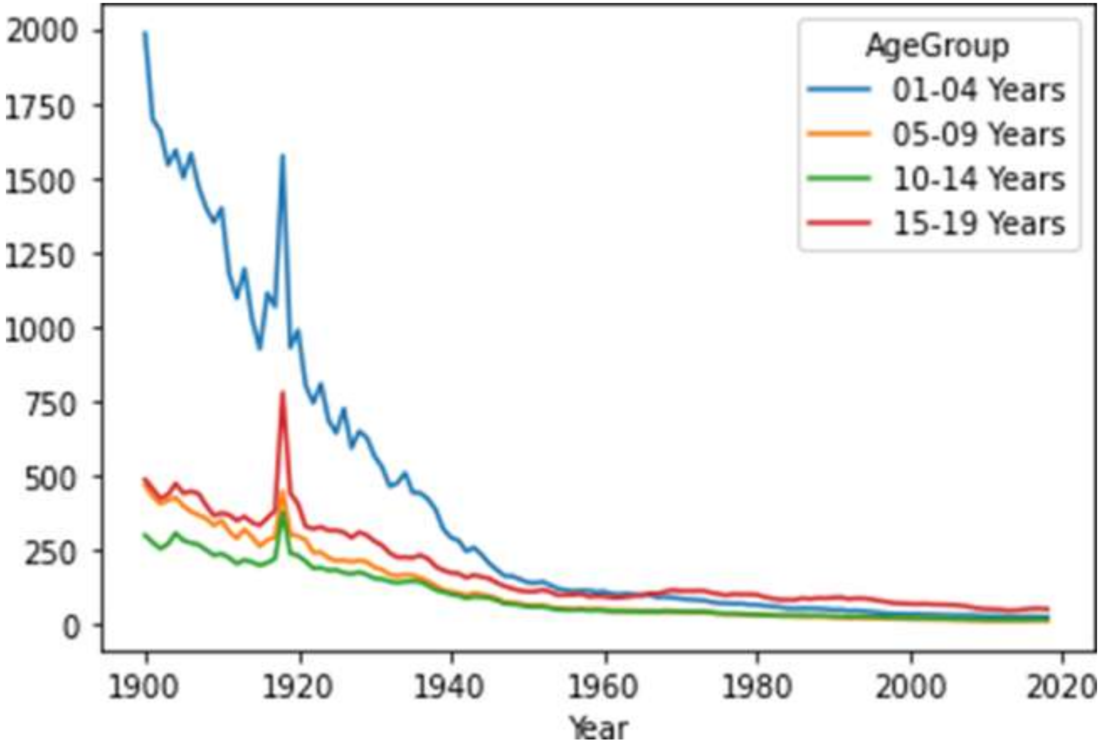
A plot() method that plots the long data with no parameters

```
mortality_data.plot()
```



A plot() method that plots the wide data with no parameters

```
mortality_wide.plot()
```



The three basic parameters for the Pandas plot() method

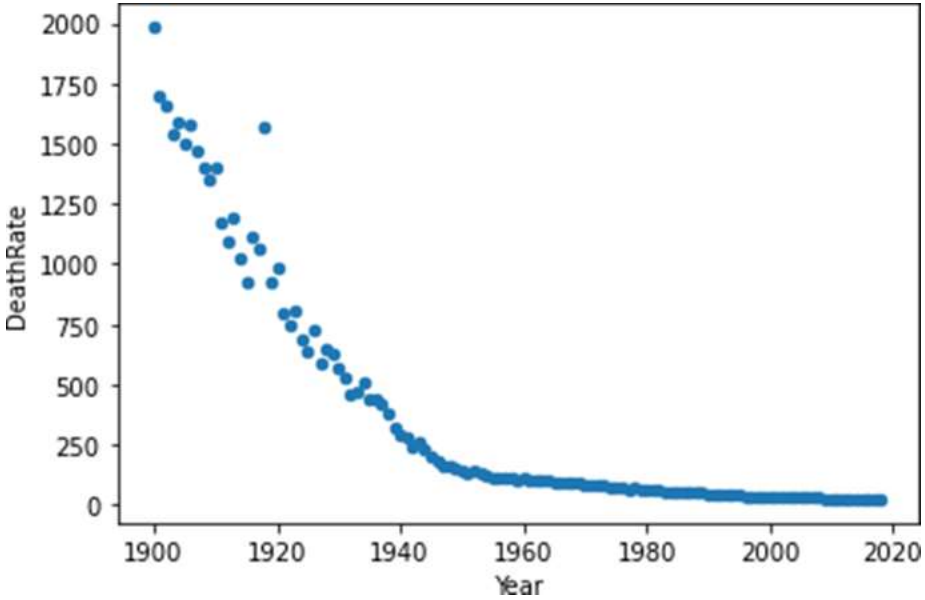
Parameter	Description
`x`	The column to be plotted on the x-axis. This column can’t be in an index.
`y`	The column or list of columns to be plotted on the y-axis.
`kind`	The kind of plot to be displayed. The default is ‘line’.

How to code the plot() method without using the kind parameter

```
mortality_data.plot.scatter()   # same as plot(kind='scatter')
mortality_data.plot.bar()       # same as plot(kind='bar')
```

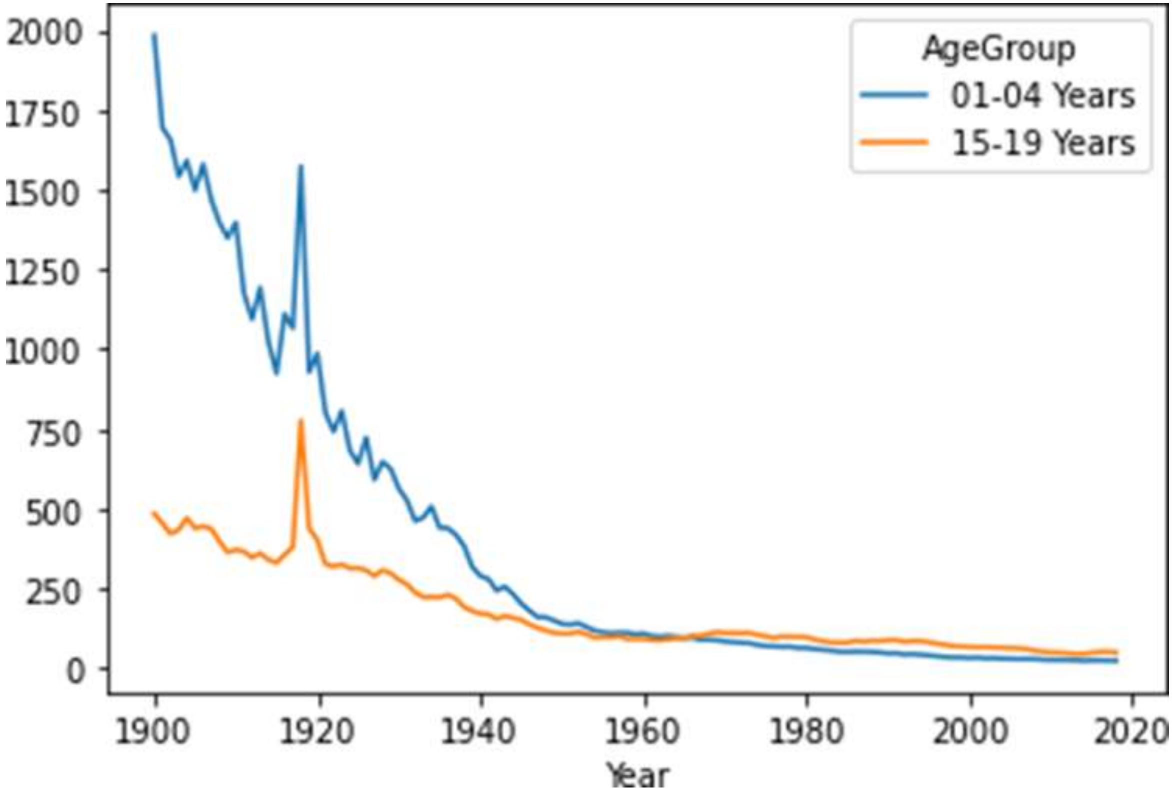
How to create a scatter plot from the long data

```
mortality_data.query('AgeGroup == "01-04 Years"') \
    .plot.scatter(x='Year', y='DeathRate')
```



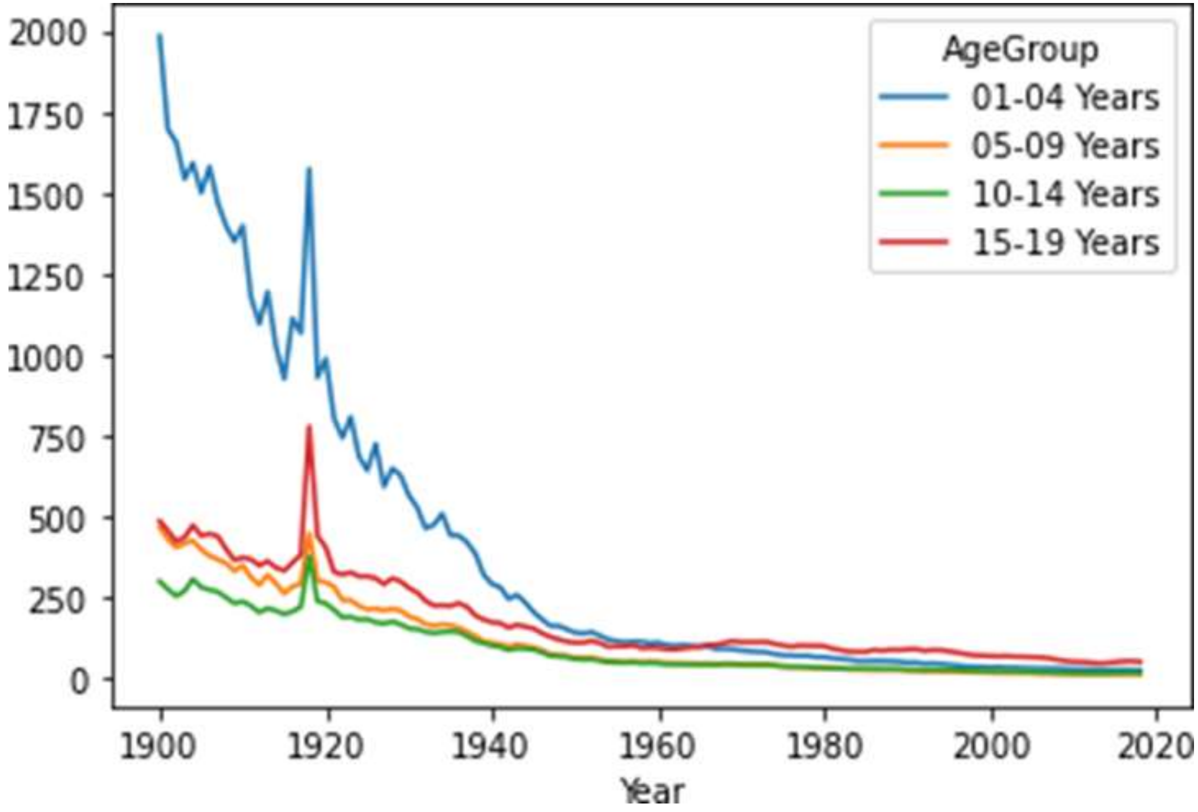
How to create a line plot from the wide data for just two of the columns

```
mortality_wide.plot.line(y=['01-04 Years','15-19 Years'])
```



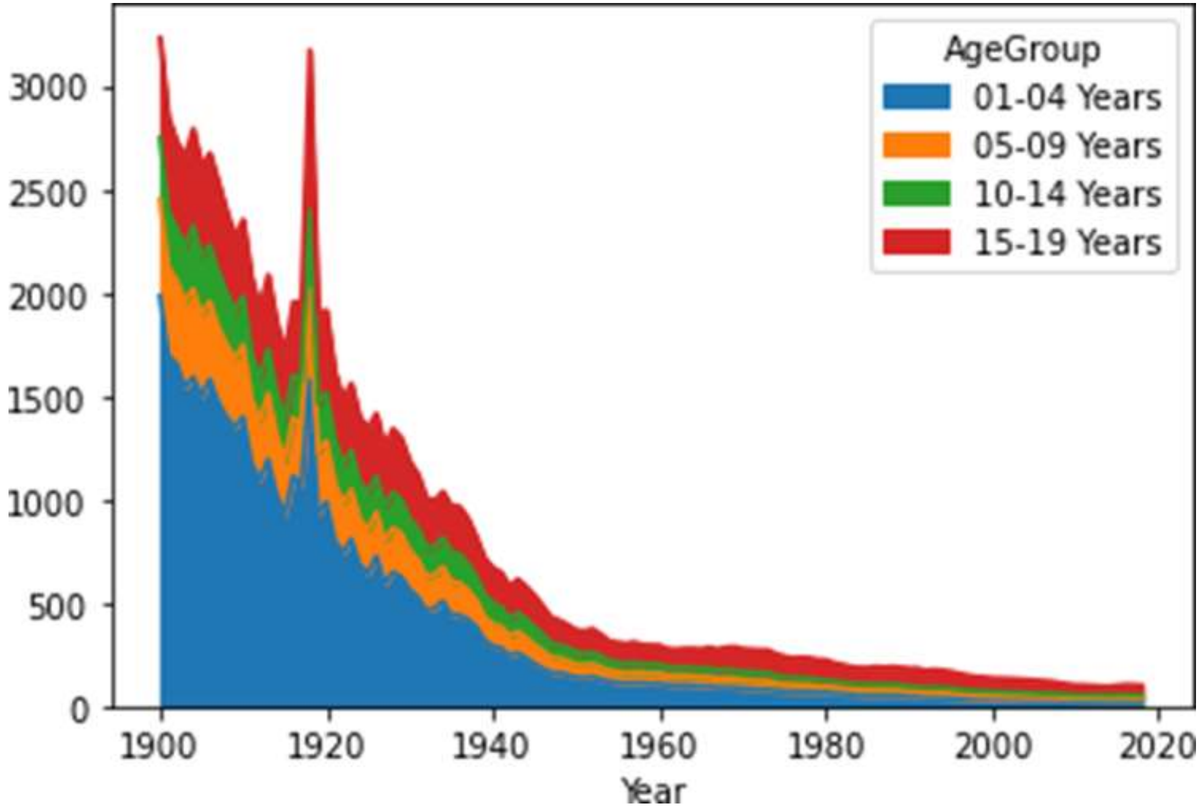
How to create a line plot from wide data

```
mortality_wide.plot.line()
```



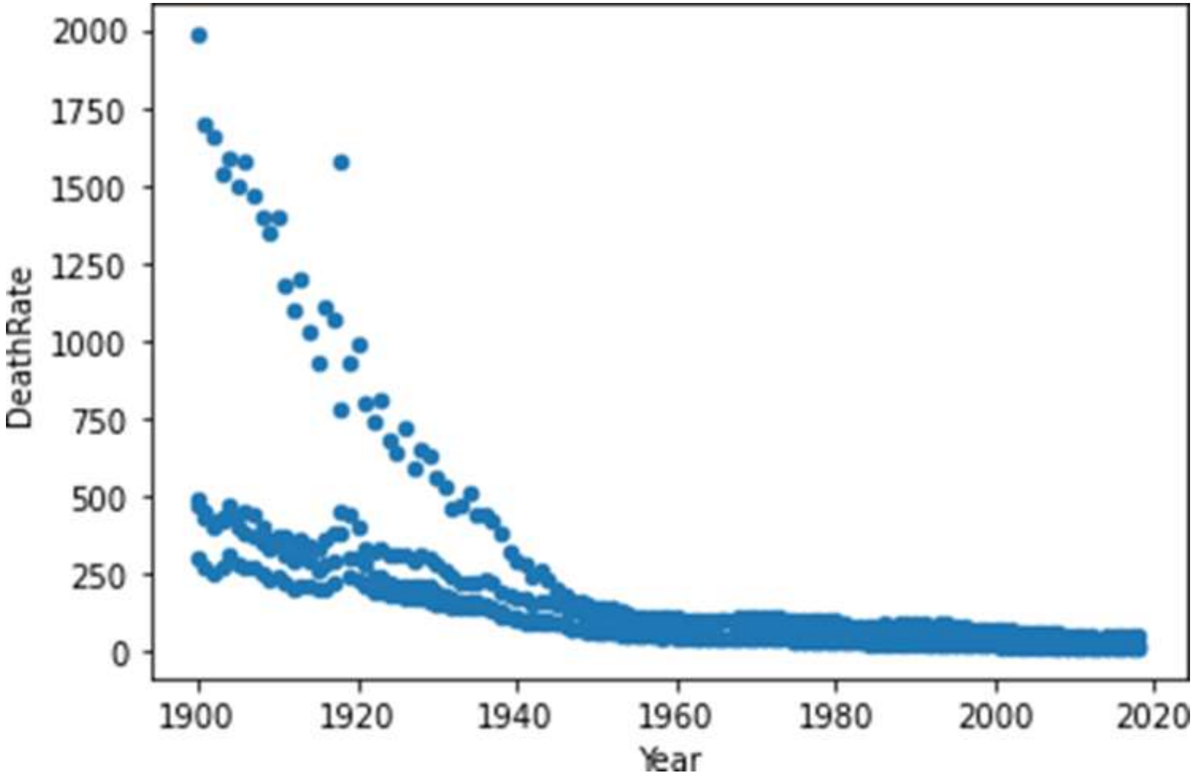
How to create an area plot from wide data

```
mortality_wide.plot.area()
```



How to create a scatter plot from long data

```
mortality_data.plot.scatter(x='Year', y='DeathRate')
```



Common problems with scatter plots based on wide data

You have to code both x and y parameters

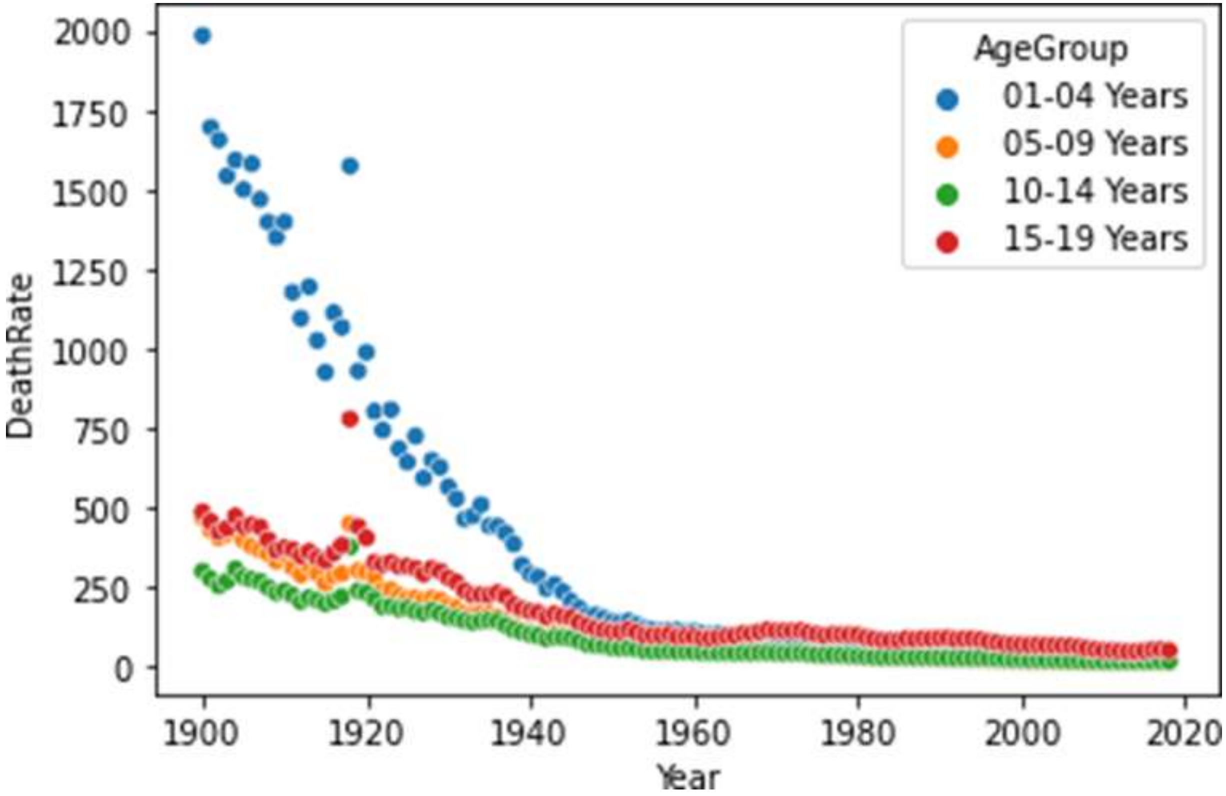
```
mortality_wide.plot.scatter()
-----------------------------
TypeError: scatter() missing 2 required positional arguments:
'x' and 'y'
```

The x parameter can’t be in an index

```
mortality_wide.plot.scatter(x='Year', y='DeathRate')
----------------------------------------------------
KeyError: 'Year'
```

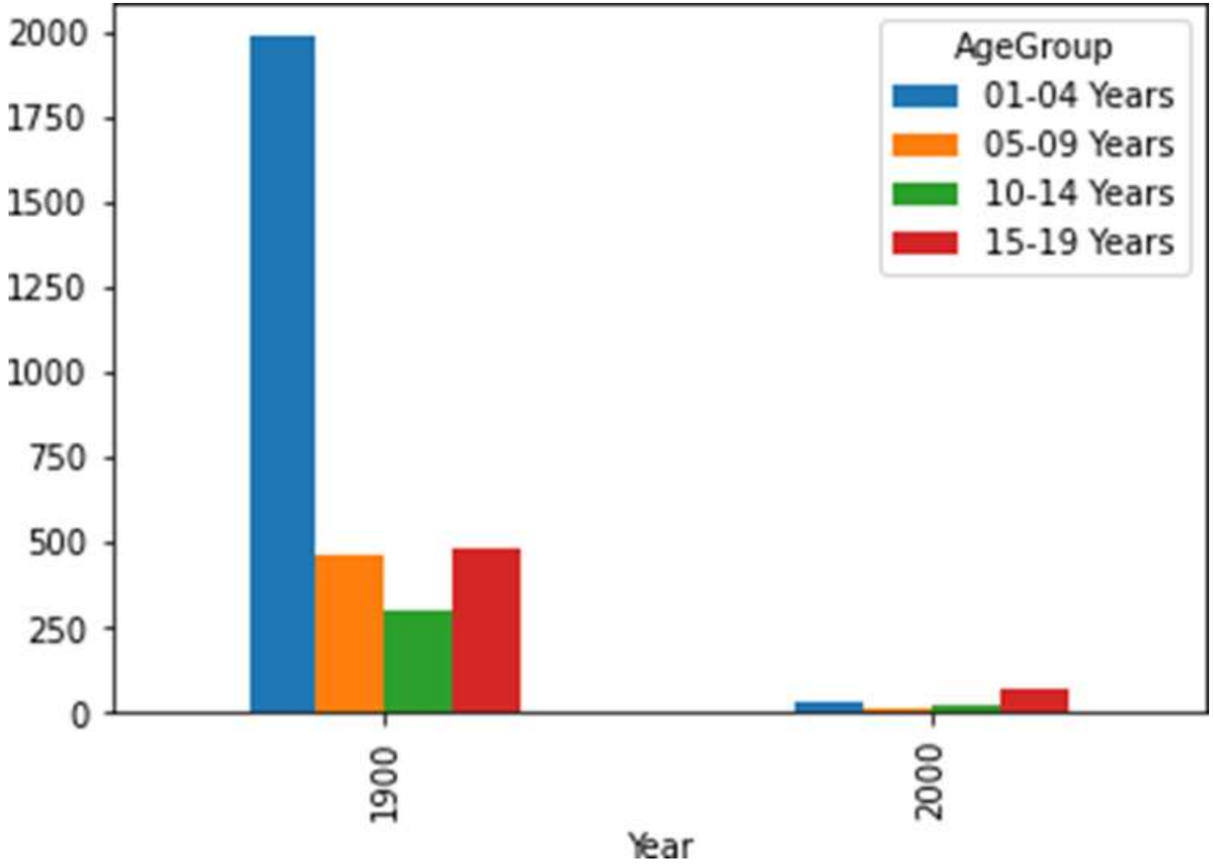
How Seaborn improves on Pandas

```
import seaborn as sns
sns.scatterplot(data=mortality_data, x='Year',
                y='DeathRate', hue='AgeGroup')
```



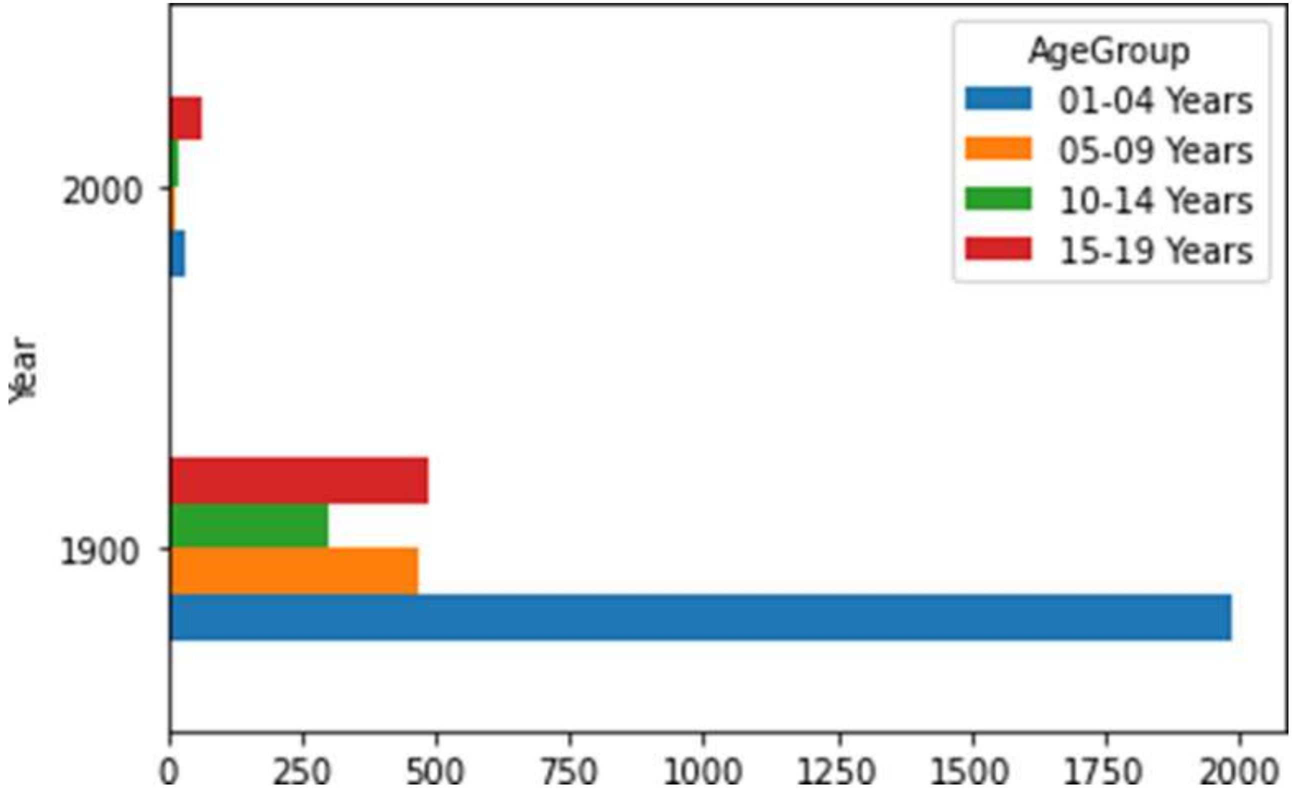
How to create a vertical bar plot from the wide data

```
mortality_wide.query('Year in (1900,2000)').plot.bar()
```



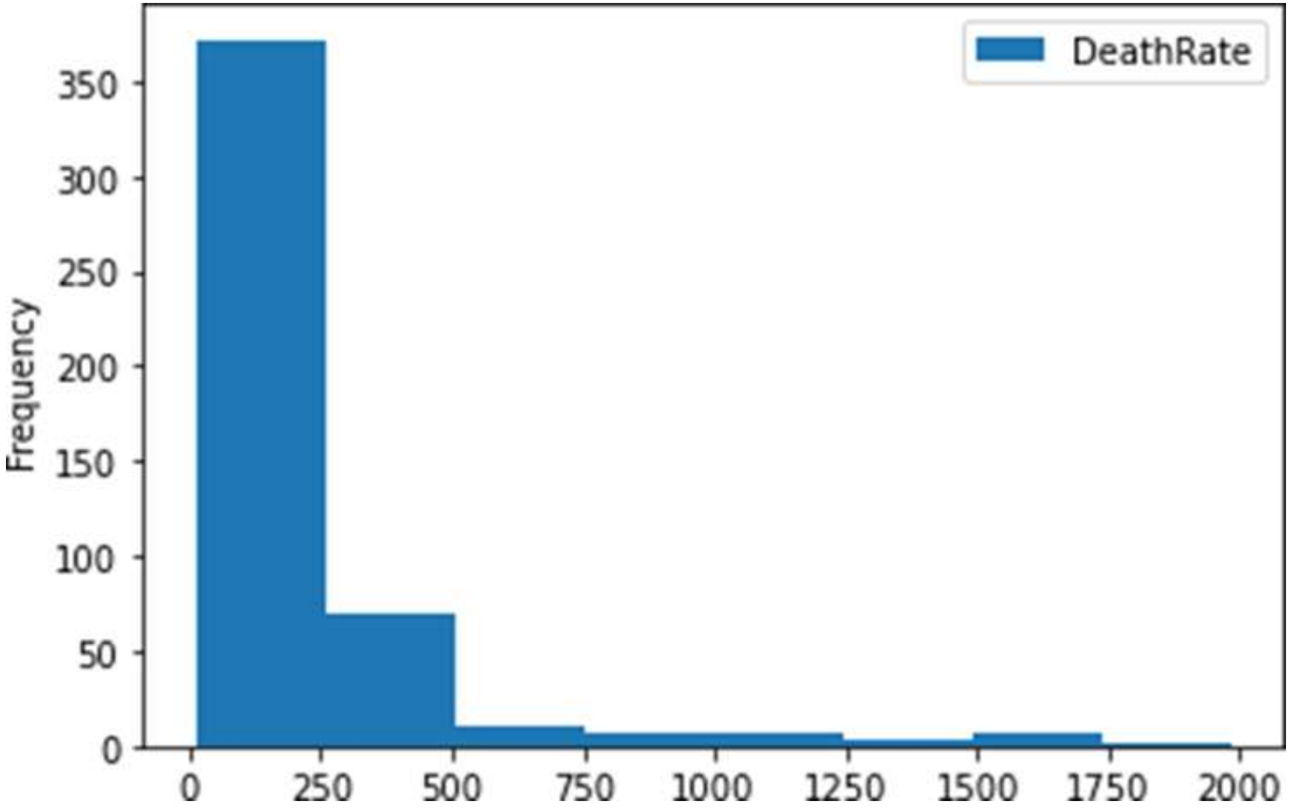
How to create a horizontal bar plot from the wide data

```
mortality_wide.query('Year in (1900,2000)').plot.barh()
```



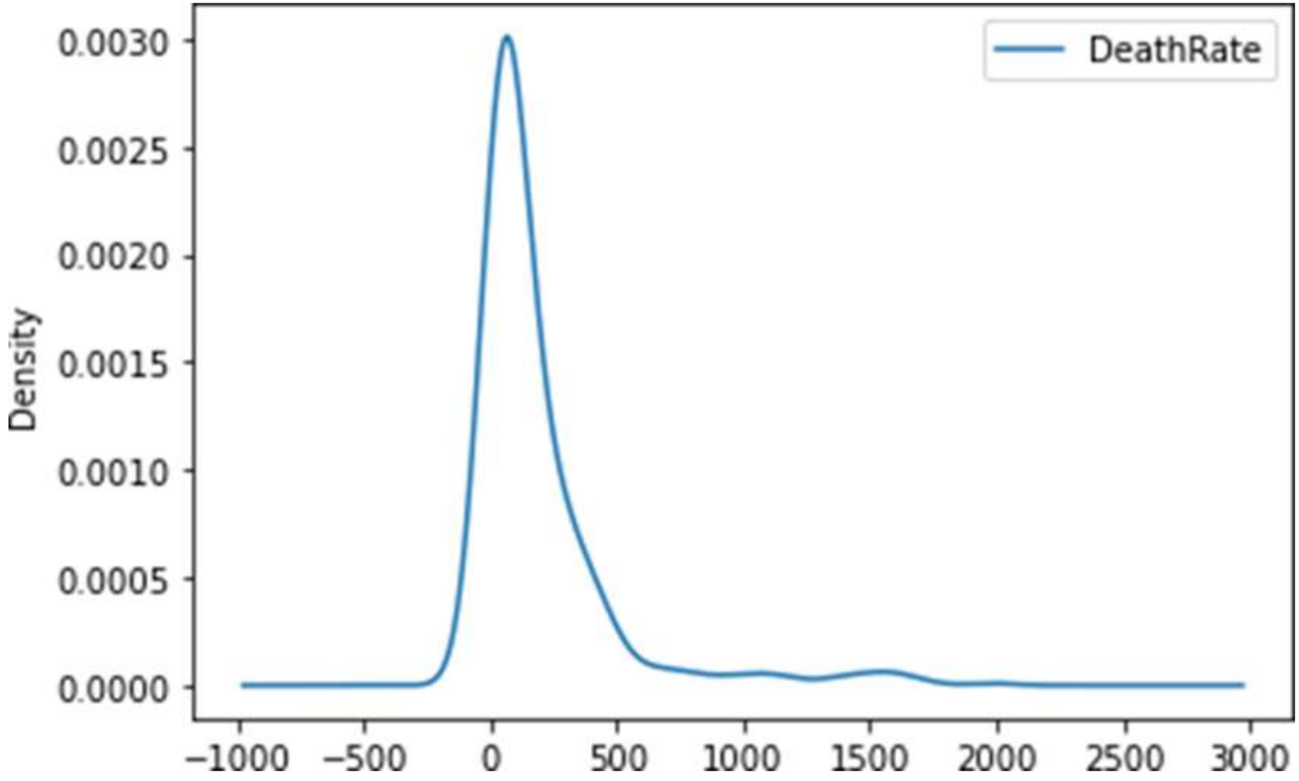
How to create a histogram

```
mortality_data.plot.hist(y='DeathRate', bins=8)
```



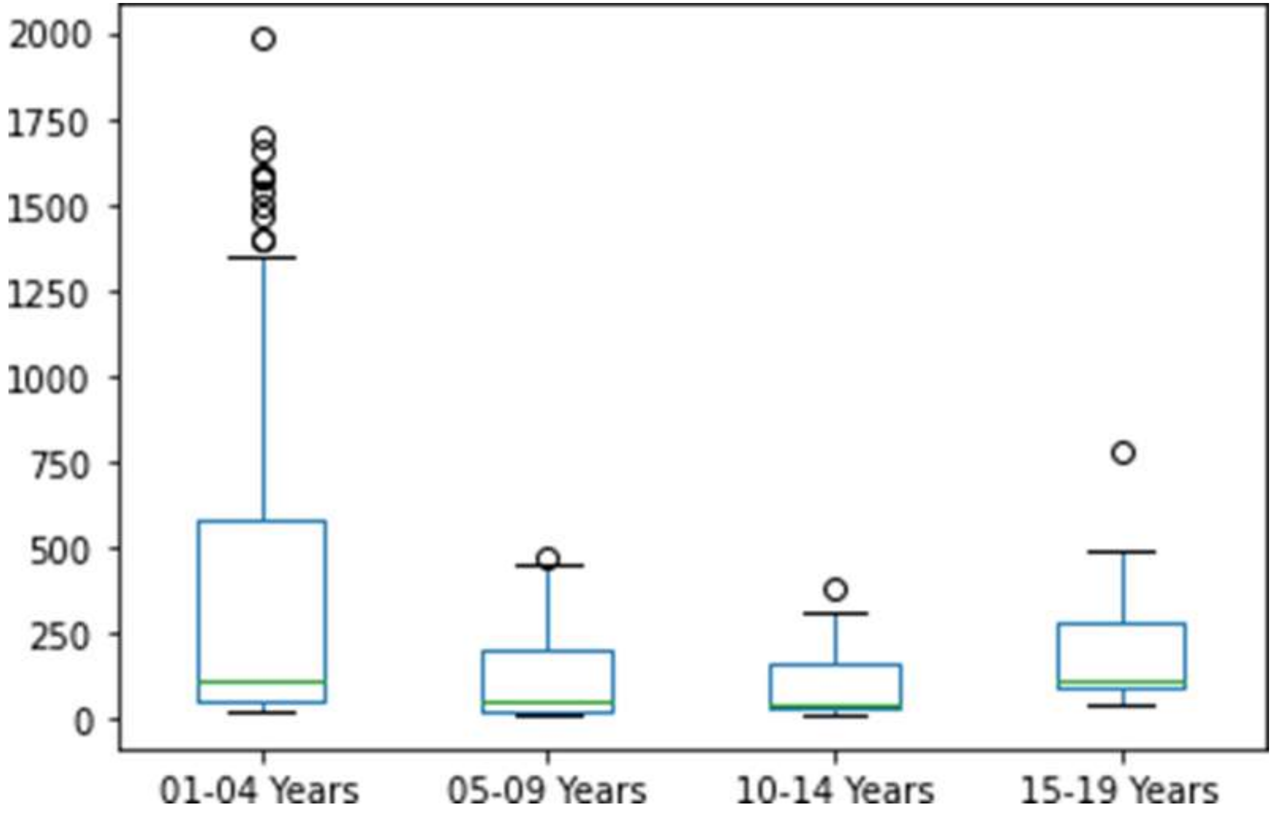
How to create a density plot

```
mortality_data.plot.density(y='DeathRate')
```



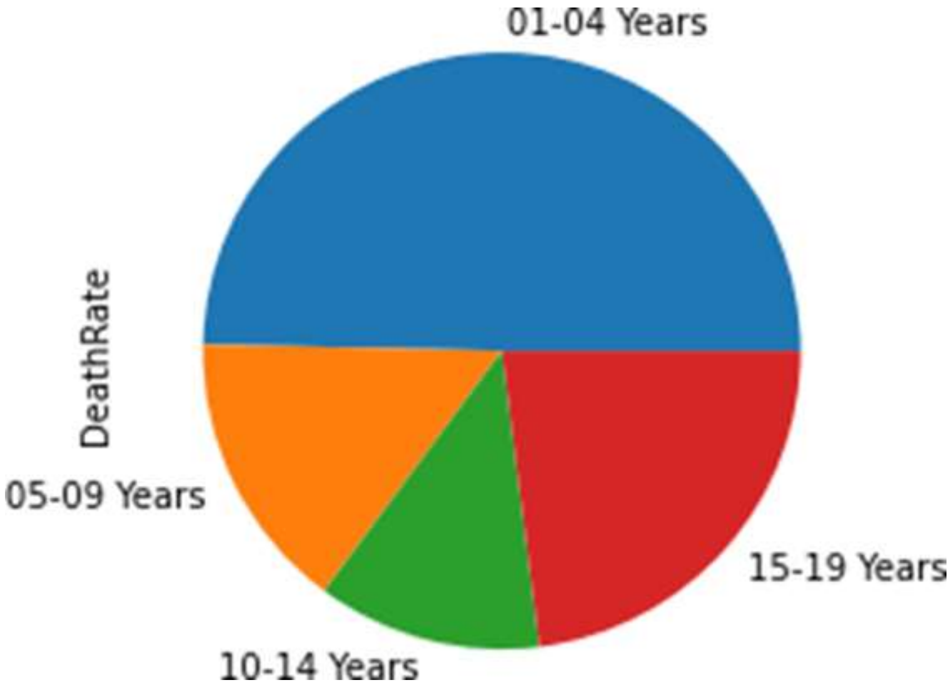
How to create a box plot from the wide data

```
mortality_wide.plot.box()
```



How to create a pie plot from the long data

```
mortality_data.groupby('AgeGroup')['DeathRate'].sum().plot.pie()
```

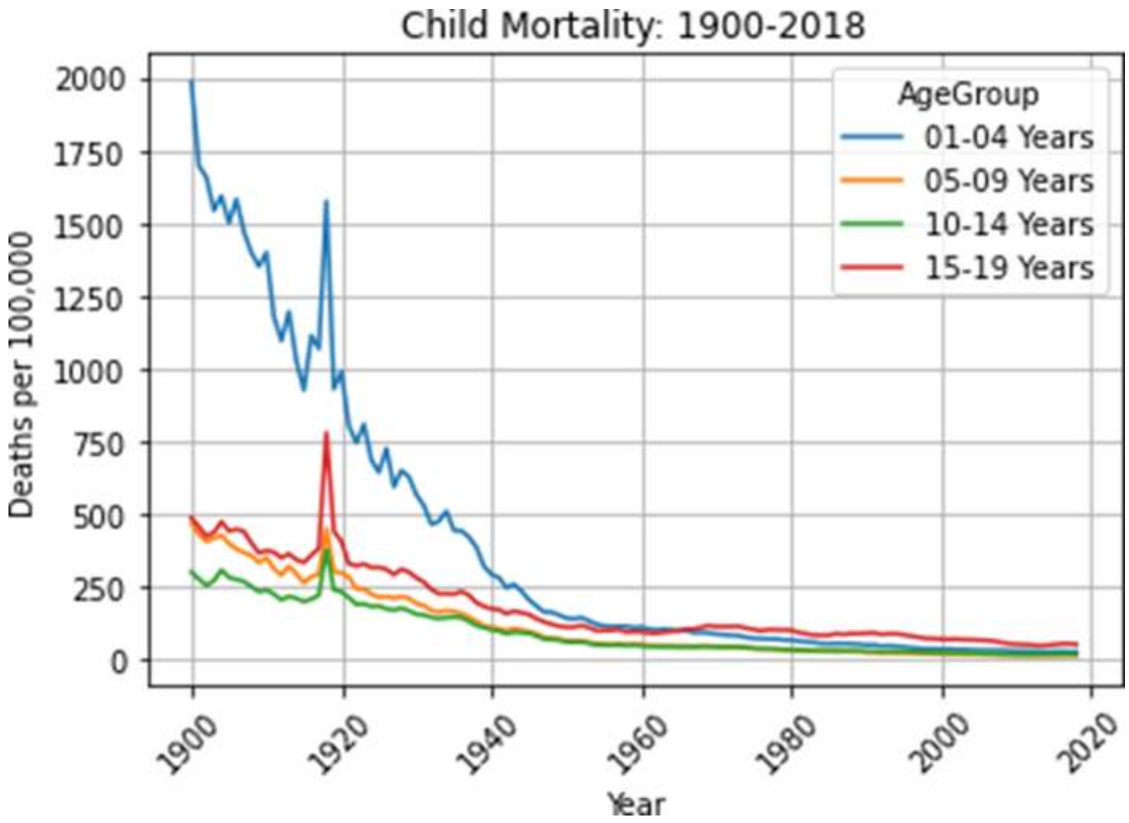


Some of the parameters for the Pandas plot() method

Parameter	Description
`title`	The title of the plot.
`legend`	If False, the legend isn't displayed. If 'reverse', the legend items are displayed in reverse order.
`grid`	If True, displays gridlines.
`rot`	The rotation of the tick labels from 0 (the default) to 360.
`xlabel, ylabel`	The label for the x- or y-axis.
`xlim, ylim`	Tuples that set the range for the x- or y-axis.
`figsize`	A tuple that sets the width and height of the plot in inches.

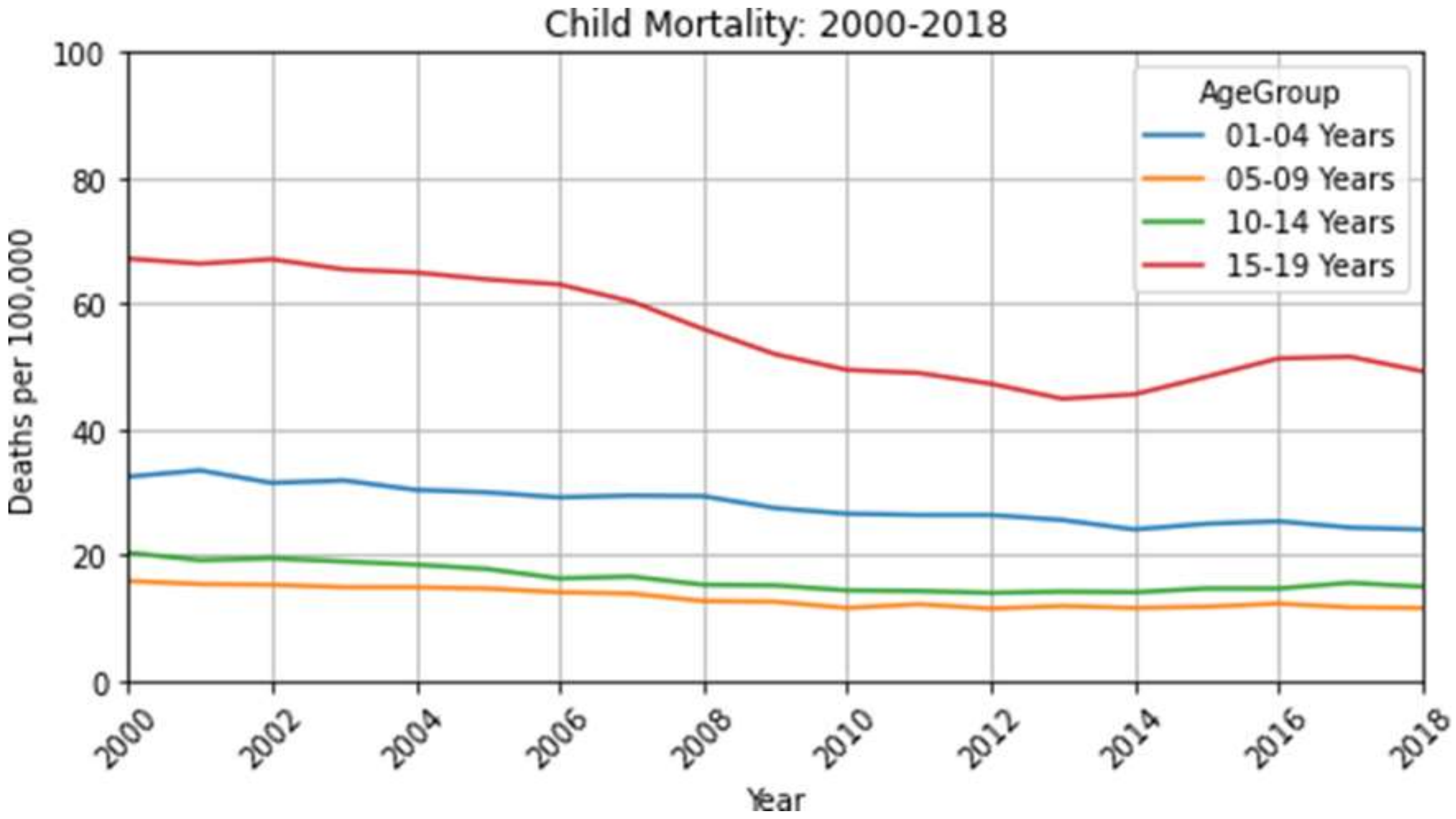
A plot with a title and a grid

```
mortality_wide.plot.line(title='Child Mortality: 1900-2018',
                         ylabel='Deaths per 100,000',
                         grid=True, rot=45)
```



A plot with x and y limits

```
mortality_wide.plot.line(title='Child Mortality: 2000-2018',
     ylabel='Deaths per 100,000', figsize=(8,4), grid=True,
     rot=45, xlim=(2000,2018), ylim=(0,100))
```



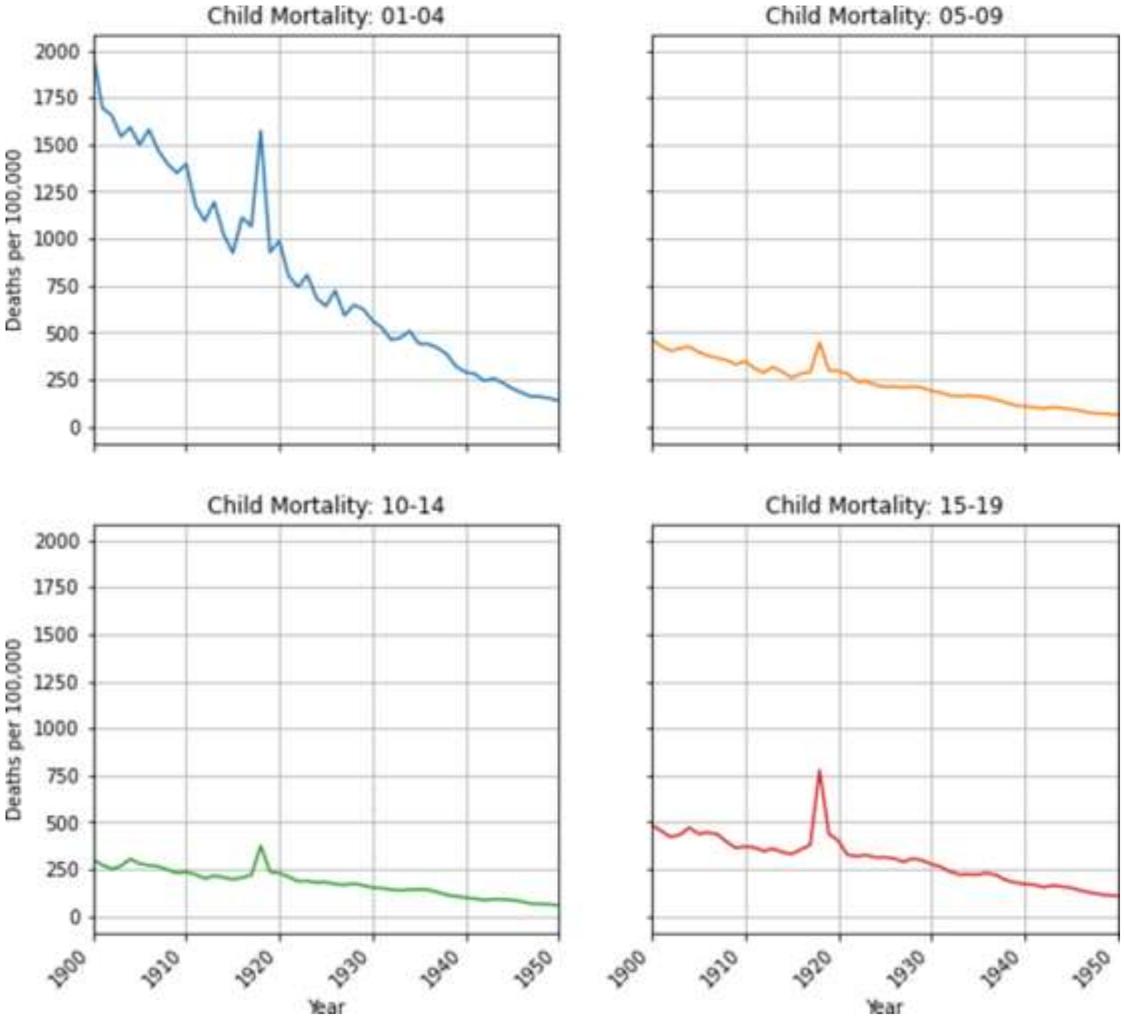
The Pandas parameters for working with subplots

Parameter	Description
`title`	A string for the plot title or a list of strings for the subplot titles.
`subplots`	If True, creates subplots if the y-axis plots more than one Series.
`layout`	A tuple that sets the number of rows and columns for the subplots.
`sharex, sharey`	If True, shares the label for the x- or y-axis so it won’t be repeated for each subplot. By default, sharex is True, sharey is False.

A plot with four subplots

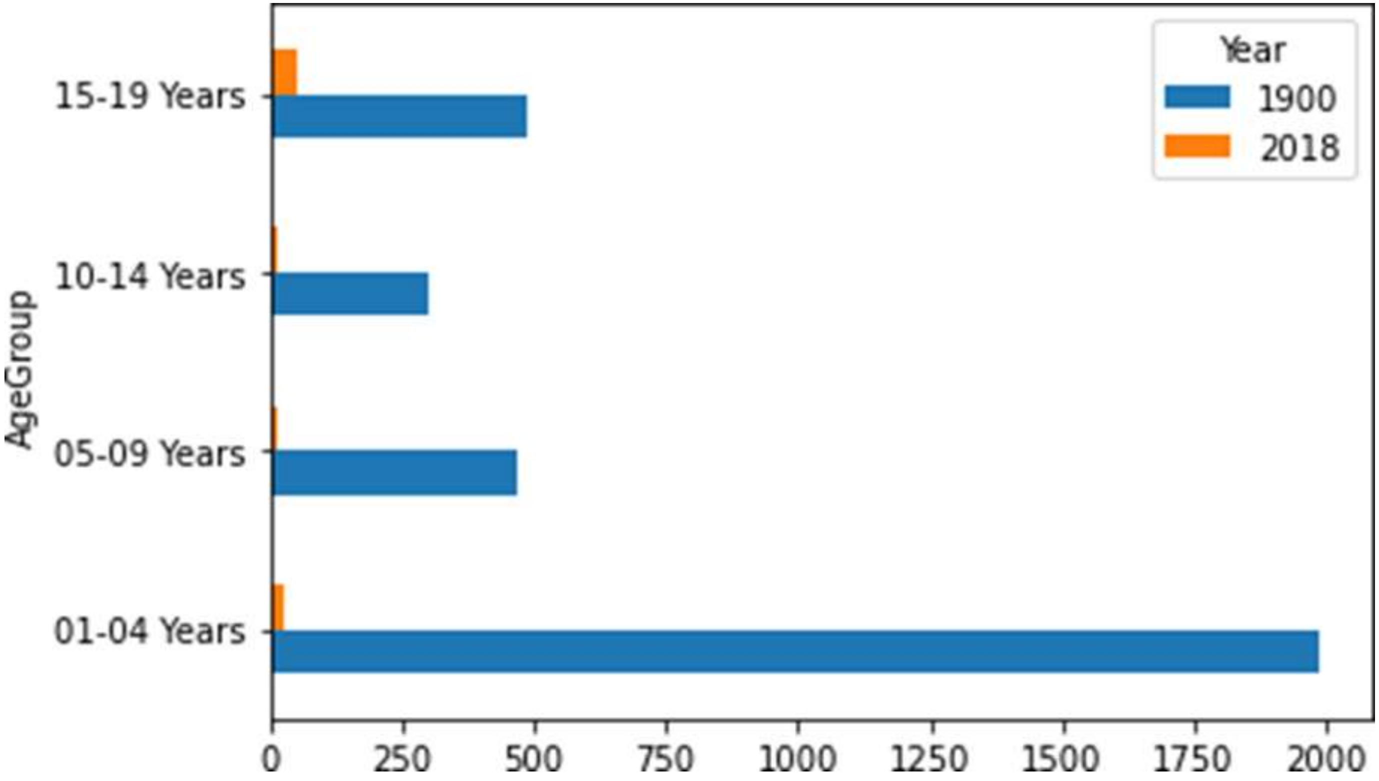
```
mortality_wide.plot.line(
    title=['Child Mortality: 01-04','Child Mortality: 05-09',
           'Child Mortality: 10-14','Child Mortality: 15-19'],
    ylabel='Deaths per 100,000', sharey=True,
    grid=True, rot=45, xlim=(1900,1950), legend=False,
    subplots=True, layout=(2,2), figsize=(10,10))
```

A plot with four subplots (continued)



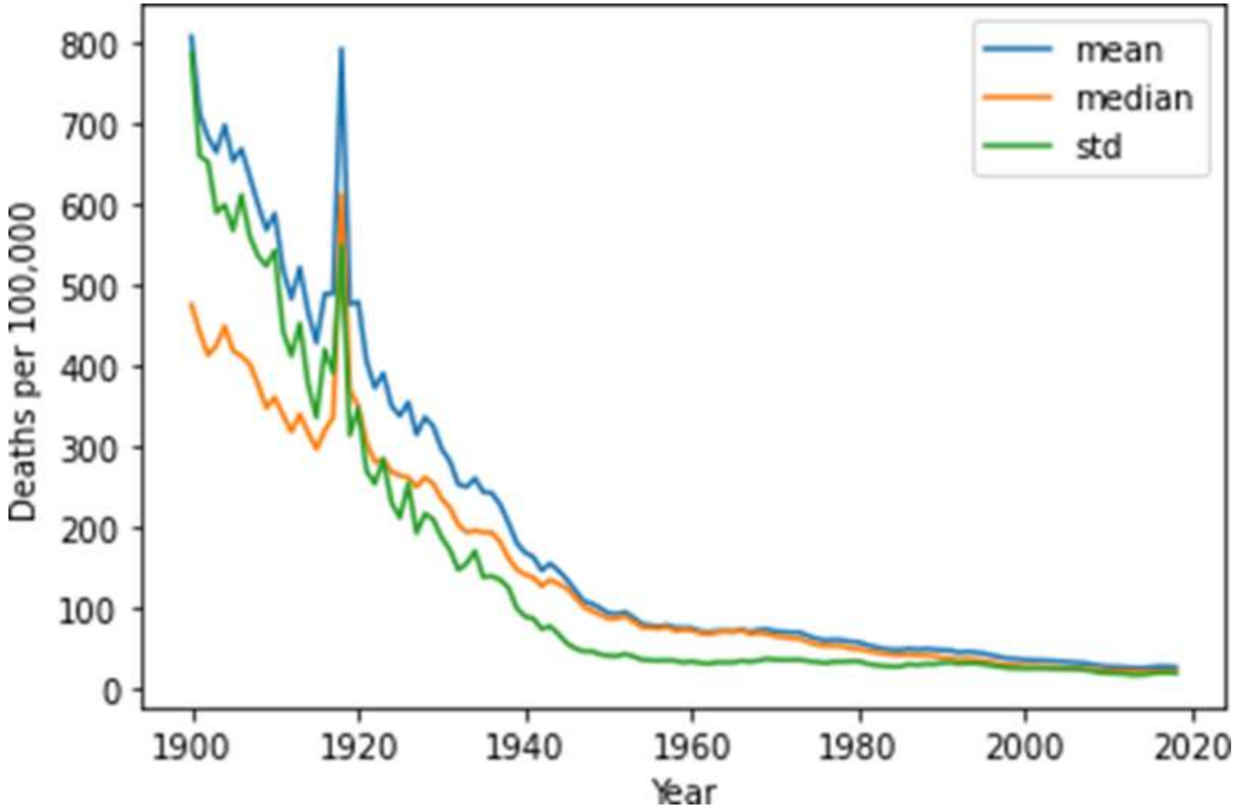
How to use chaining to create a bar plot from long data

```
mortality_data.query('Year in (1900,2018)') \
     .pivot(index='AgeGroup', columns='Year',
            values='DeathRate').plot.barh()
```



How to use the groupby() method and chaining to create a plot

```
mortality_data.groupby('Year')['DeathRate'] \
    .agg(['mean','median','std']) \
    .plot(ylabel='Deaths per 100,000')
```


mean
median
std
Deaths per 100,000
Year