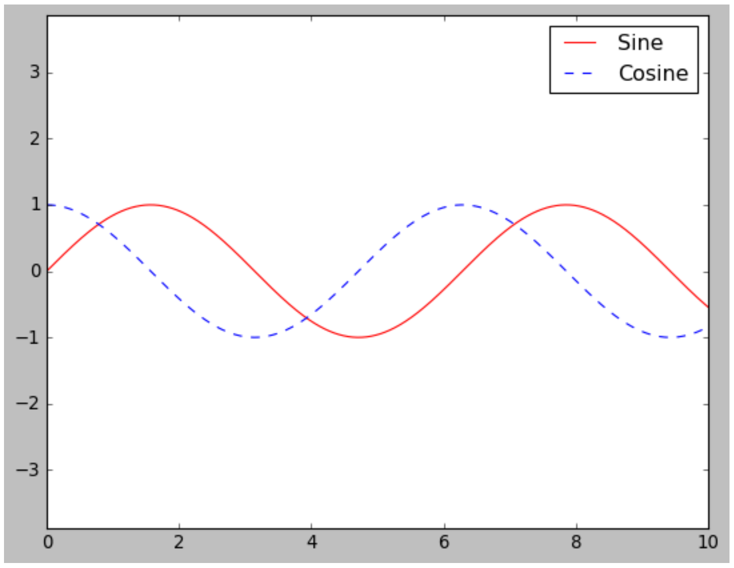
In [3]:
```
import matplotlib.pyplot as plt
plt.style.use('classic')
```

In [4]:
```
%matplotlib inline
import numpy as np
```

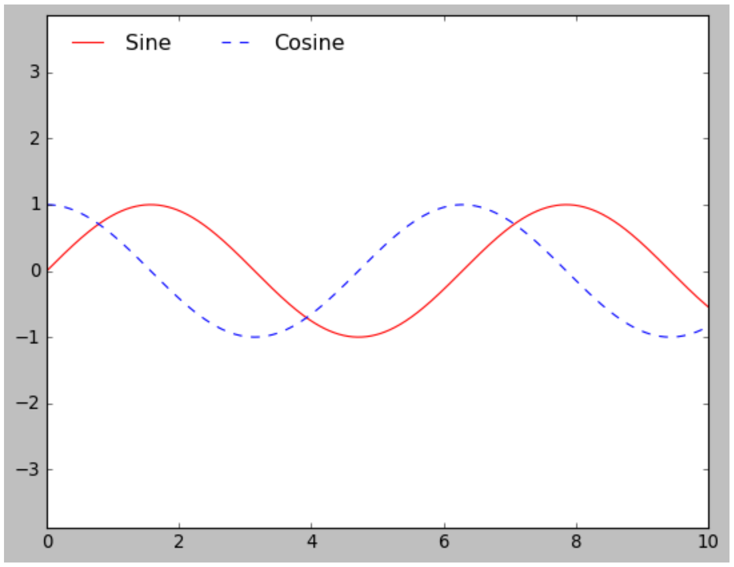
In [5]:
```
#creating legend with plt.legend()
x=np.linspace(0,10,1000)
fig=plt.figure()
ax=plt.axes()
ax.plot(x,np.sin(x),"-r",label="Sine")
ax.plot(x,np.cos(x),"--b",label="Cosine")
ax.axis("equal")
ax.legend()
```

Out[5]:
```
<matplotlib.legend.Legend at 0x1dac14741c0>
```



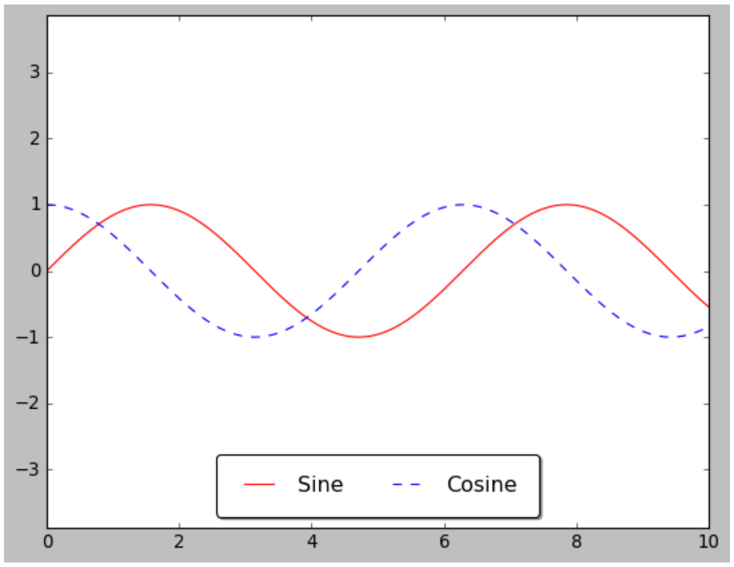
In [6]:
```
#turning off frame and location
#frameon to remove or keep box around legend,loc is used for specifying location,ncol is used for representing number of columns
ax.legend(ncol=2,loc="upper left",frameon=False)
fig
```

Out[6]:



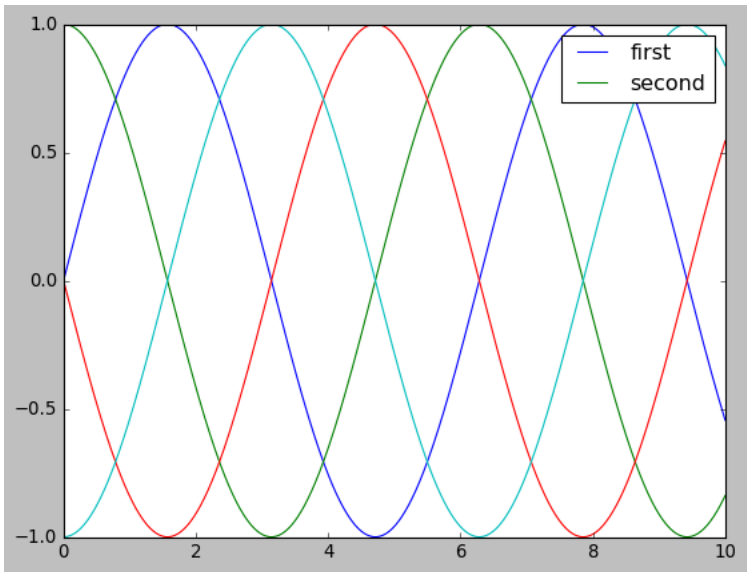
In [7]:
```
#adding shadow,fancy boxes,changing transperency of edge of legend box and adding padding to text
ax.legend(fancybox=True,ncol=2,loc="lower center",framealpha=1,shadow=True,borderpad=1)
fig
```

Out[7]:



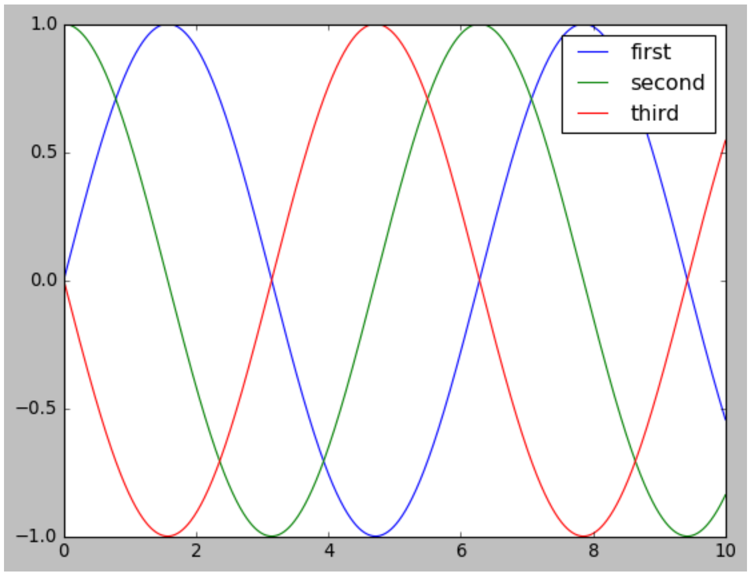
In [8]:
```
#changing index of legends by obtaining list of indexes
y=np.sin(x[:, np.newaxis]+np.pi*np.arange(0,2,0.5))
lines=plt.plot(x,y)#list of plot.2D instances
plt.legend(lines[0:2],["first","second"])
```

Out[8]:
```
<matplotlib.legend.Legend at 0x1dabfe8f490>
```



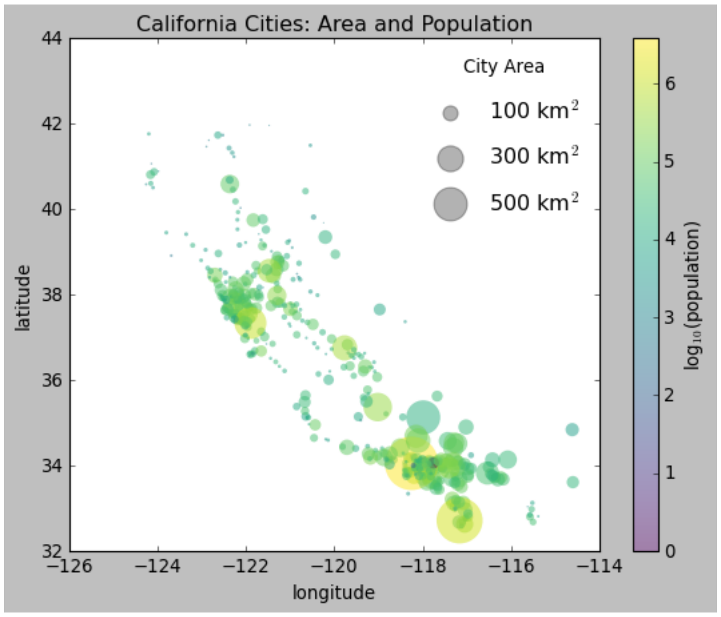
In [9]:
```
#another method
plt.plot(x,y[:,0],label="first")#[:,0]->first dimension is skipped next dimension first element
plt.plot(x,y[:,1],label="second")
plt.plot(x,y[:,2],label="third")
plt.legend()#legend ignores attributes without labels be default
```

Out[9]:
```
<matplotlib.legend.Legend at 0x1dac15f3130>
```



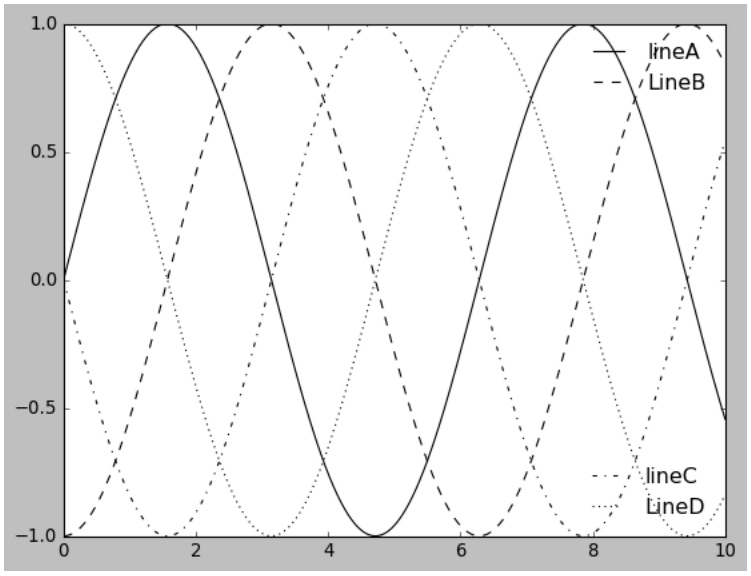
In [13]:
```
#legends for size of points
import pandas as pd
cities=pd.read_csv("california_cities.csv")
#extracting particular data
lat,lon=cities["latd"],cities["longd"]#getting latitude and longitude arrays
population,area=cities["population_total"],cities["area_total_km2"]#getting population and area arrays
plt.scatter(lon,lat,label=None,c=np.log10(population),s=area,linewidth=0,alpha=0.5,cmap="viridis")
plt.axis()#making aspect ratio as equal
plt.xlabel("longitude")#labelling x axis as longitude
plt.ylabel("latitude")#labelling y axis as latitude
plt.colorbar(label="log$_{10}$(population)")#for printing log10 in a precise way
plt.title("California Cities: Area and Population");
for area in [100,300,500]:#creating points for 3 values of areas
    plt.scatter([],[], c='k',alpha=0.3,s=area,label=str(area)+" km$^2$")#creating legend objects by plotting with empty lsits
plt.legend(scatterpoints=1,frameon=False,labelspacing=1,title="City Area")#labelspacing attribute for space between labels
#scatterpoints value is 1 as we want only one point to rpresent values in legend
```

Out[13]:
```
<matplotlib.legend.Legend at 0x1dac2873790>
```



In [15]:
```
#adding multiple legends
fig,ax=plt.subplots()
lines=[]
styles=["-","--","-.",":"]
x=np.linspace(0,10,10000)
for i in range(4):
    lines+=ax.plot(x,np.sin(x-i*np.pi/2),styles[i],color="black")
ax.legend(lines[:2],["lineA","LineB"],loc="upper right",frameon=False)
from matplotlib.legend import Legend
#creating second legend manually
leg=Legend(ax,lines[2:],["lineC","LineD"],loc="lower right",frameon=False)
ax.add_artist(leg)#lower level function add artist will add new legend
```

Out[15]:
```
<matplotlib.legend.Legend at 0x1dac2cc80d0>
```


lineA
LineB
lineC
LineD