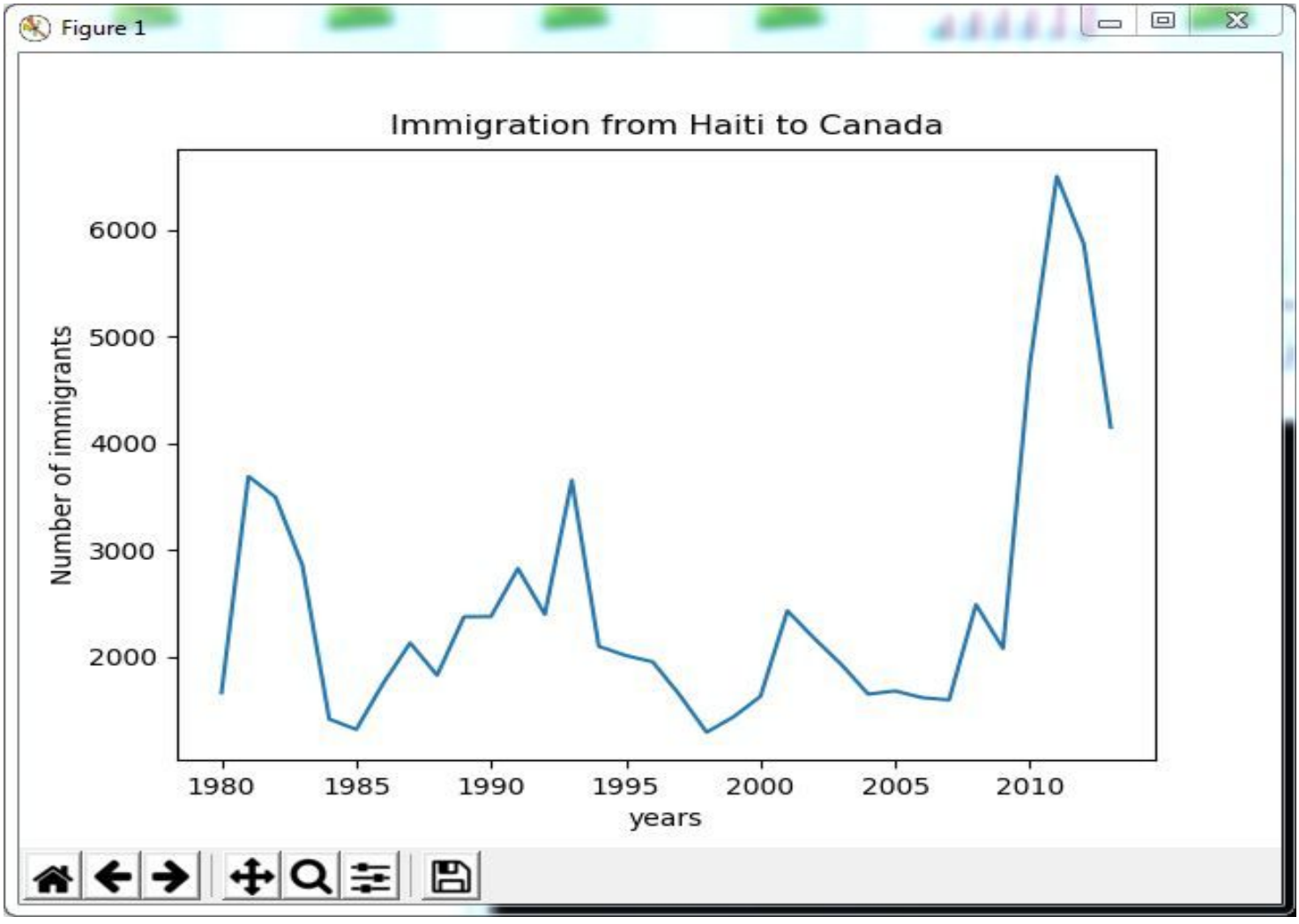
1. Line Plot (Immigration from Haiti to Canada):

Python Code:

```
import pandas as pd
import matplotlib.pyplot as plt
df_can=pd.read_excel(
        'https://ibm.box.com/shared/static/lw190pt9zpy5bd1ptyg2aw15awomz9pu.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
df_can.set_index('OdName',inplace=True)
years=list(map(int,range(1980,2014)))
print(df_can)
print()
print(df_can.loc['Haiti',years])
df_can.loc['Haiti',years].plot(kind="line")
plt.title("Immigration from Haiti to Canada")
plt.xlabel('years')
plt.ylabel('Number of immigrants')
plt.show()
```

Output:

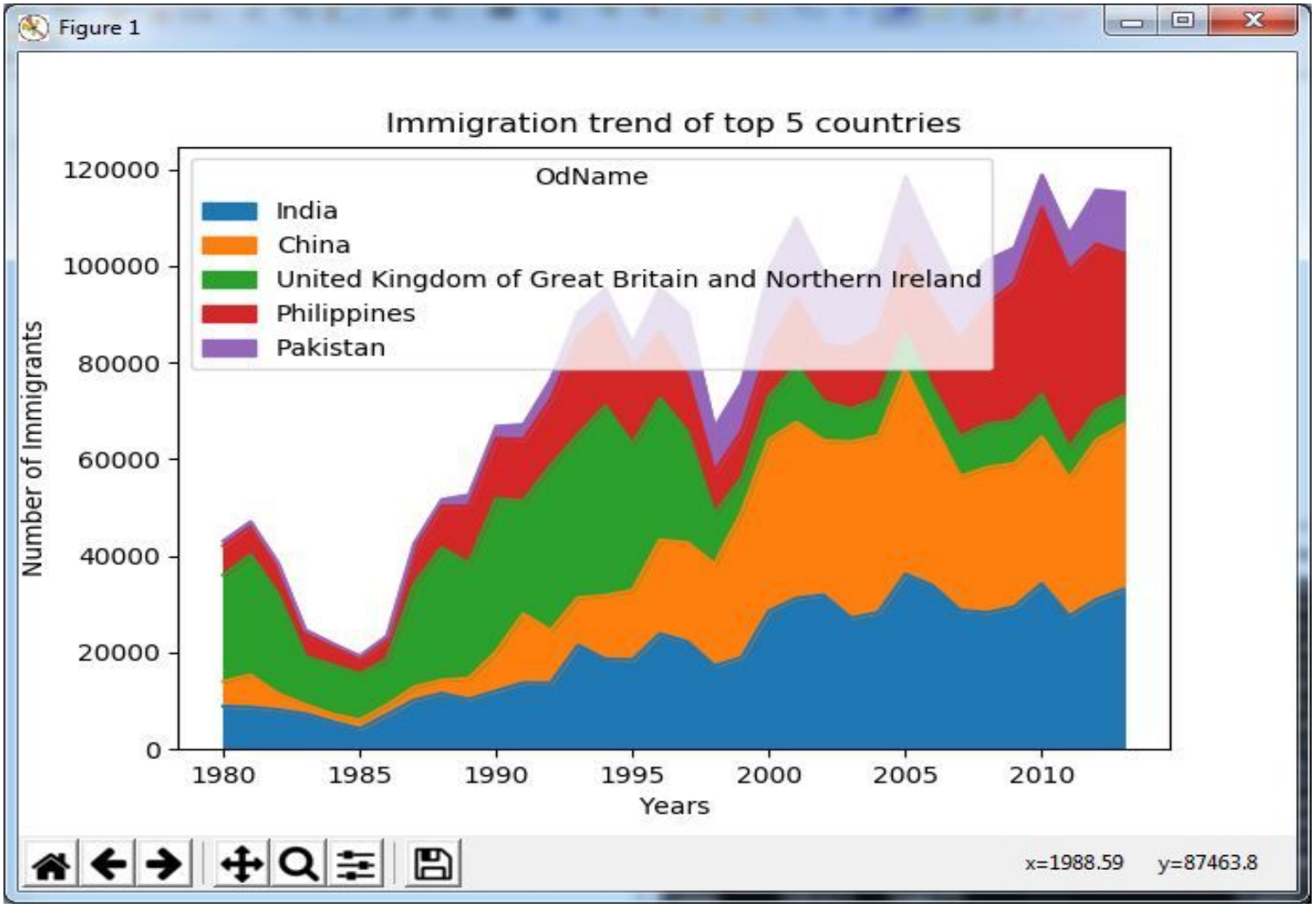


2. Area Plot (Immigration trend of top 5 countries):

Python Code:

```
import pandas as pd
import matplotlib.pyplot as plt
df_can=pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
years=list(map(int,range(1980,2014)))
df_can.set_index('OdName',inplace=True)
df_can.sort_values(['Total'],ascending=False,axis=0,inplace=True)
df_top5=df_can.head()
df_top5=df_top5[years].transpose()
df_top5.plot(kind="area")
plt.title("Immigration trend of top 5 countries")
plt.xlabel("Years")
plt.ylabel("Number of Immigrants")
plt.show()
```

Output:

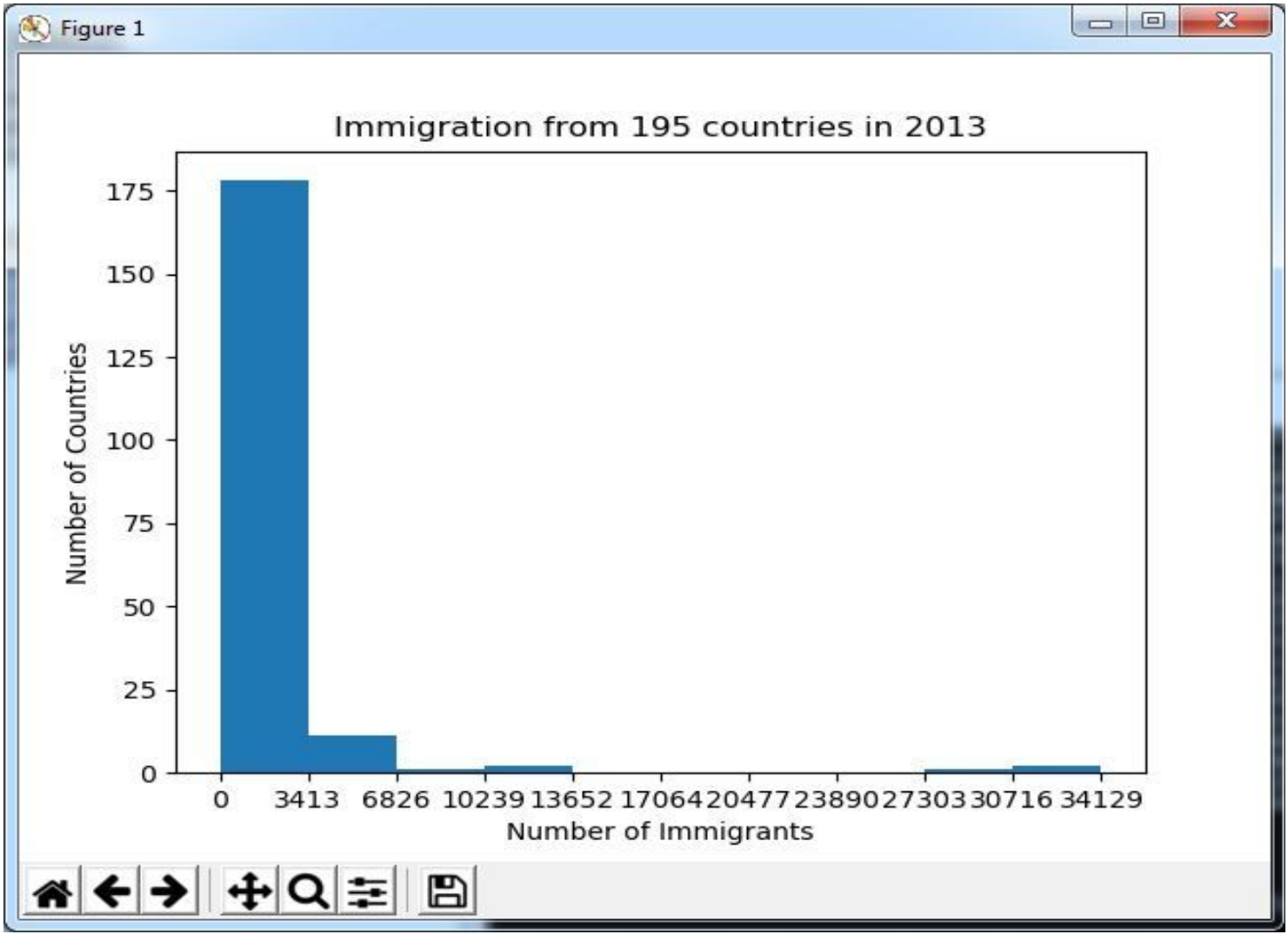


3. Histogram (Immigration from 195 countries in 2013):

Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df=pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)

hist,bin_edges = np.histogram(df[2013])
df[2013].plot(kind='hist',xticks=bin_edges)
plt.title("Immigration from 195 countries in 2013")
plt.xlabel("Number of Immigrants")
plt.ylabel("Number of Countries")
plt.show()
```

Output:

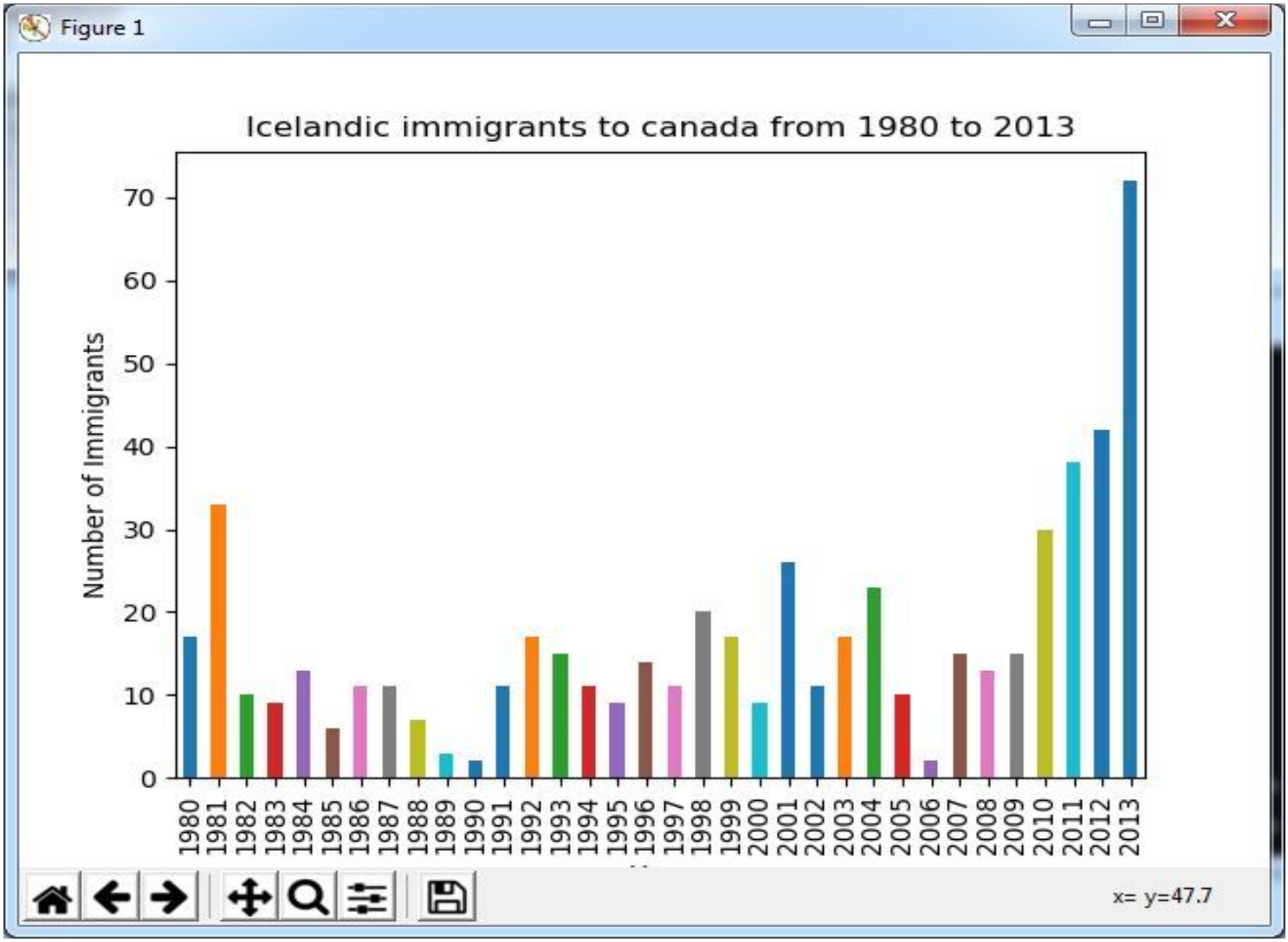


4. Bar Chart (Icelandic immigrants to canada from 1980 to 2013):

Python Code:

```
import matplotlib.pyplot as plt
import pandas as pd

df_canada=pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
df_canada.set_index('OdName',inplace=True)
years = list(map(int,range(1980,2014)))
df=df_canada.loc['Iceland',years]
df.plot(kind='bar')
plt.title("Icelandic immigrants to canada from 1980 to 2013")
plt.xlabel("Year")
plt.ylabel("Number of Immigrants")
plt.show()
```

Output:

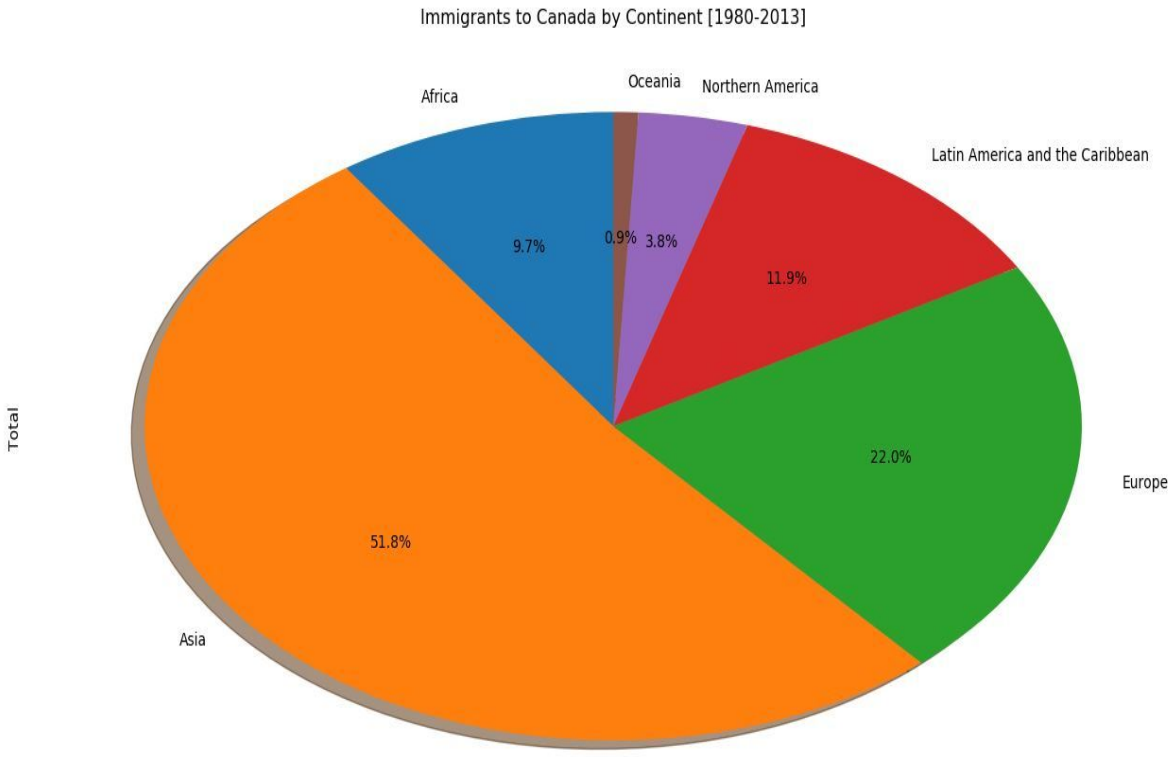


5. Pie Chart (Immigrants to Canada by Continent [1980-2013]):

Python Code:

```
import matplotlib.pyplot as plt
import pandas as pd

df_can = pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
df=df_can.groupby('AreaName',axis=0).sum()
print(df)
df['Total'].plot(
        kind='pie',
        autopct='%1.1f%%',
         shadow=True,
        #labels=None,
        startangle=90,
        #pctdistance=1.2
)
#plt.legend(labels=df.index,loc='upper left')
plt.title("Immigrants to Canada by Continent [1980-2013]")
plt.show()
```

Output:

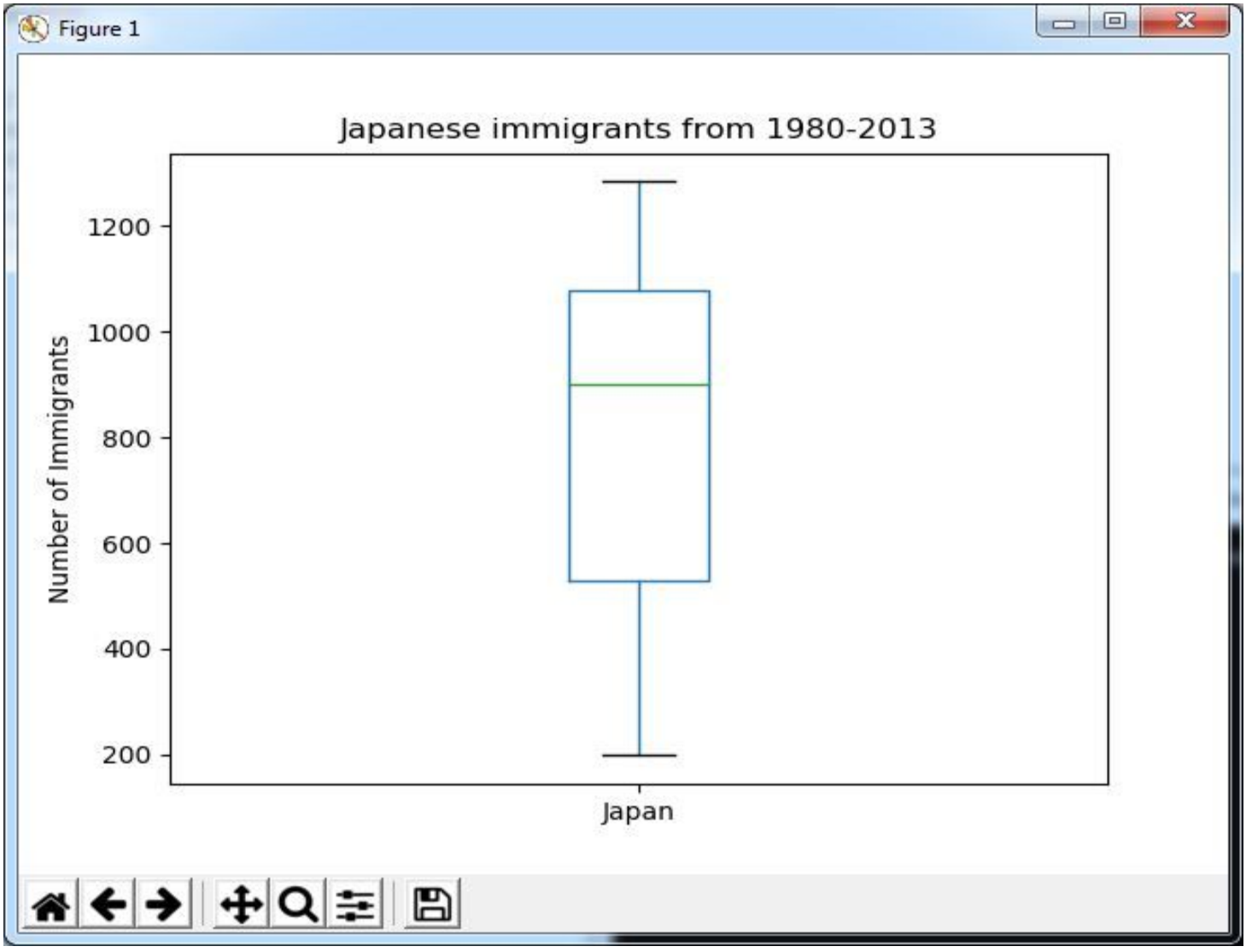


6. Box Plot (Japanese immigrants from 1980-2013):

Python Code:

```
import matplotlib.pyplot as plt
import pandas as pd

df_can=pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
df_can.set_index('OdName',inplace=True)
years=list(map(int,range(1980,2014)))
df_jap=df_can.loc['Japan',years]
df_jap.plot(kind='box')
plt.title("Japanese immigrants from 1980-2013")
plt.ylabel('Number of Immigrants')
plt.show()
```

Output:

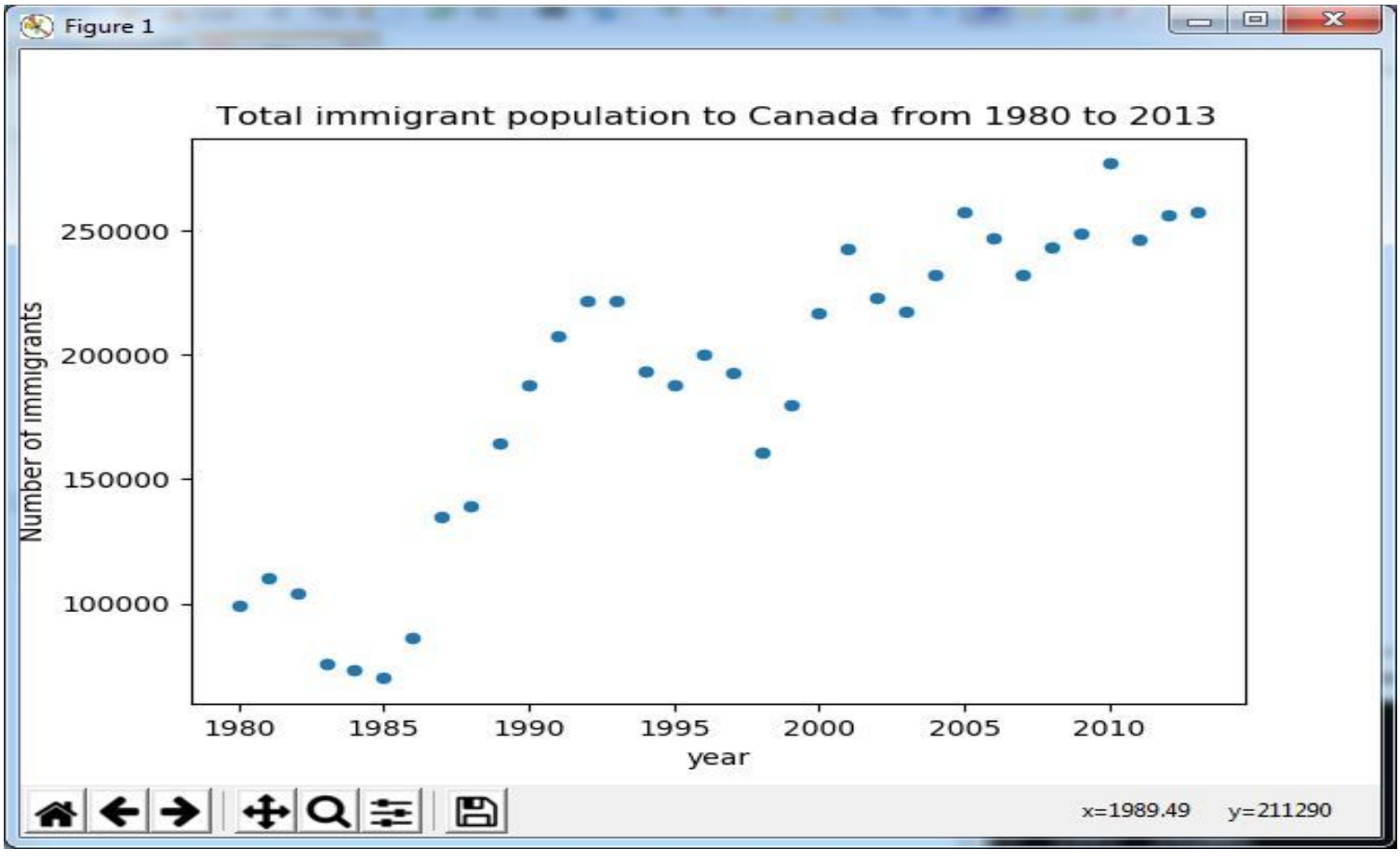


7. Scatter Plot (Total immigrant population to Canada from 1980 to 2013):

Python Code:

```
import matplotlib.pyplot as plt
import pandas as pd
df_can = pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
years=list(map(int,range(1980,2014)))
df_tot = pd.DataFrame(df_can[years].sum())
df_tot.reset_index(inplace = True)
df_tot.columns = ['year', 'total']
df_tot.plot(
        kind='scatter',
        x='year',
        y='total'
)
plt.title("Total immigrant population to Canada from 1980 to 2013")
plt.xlabel("year")
plt.ylabel("Number of immigrants")
plt.show()
```

Output:

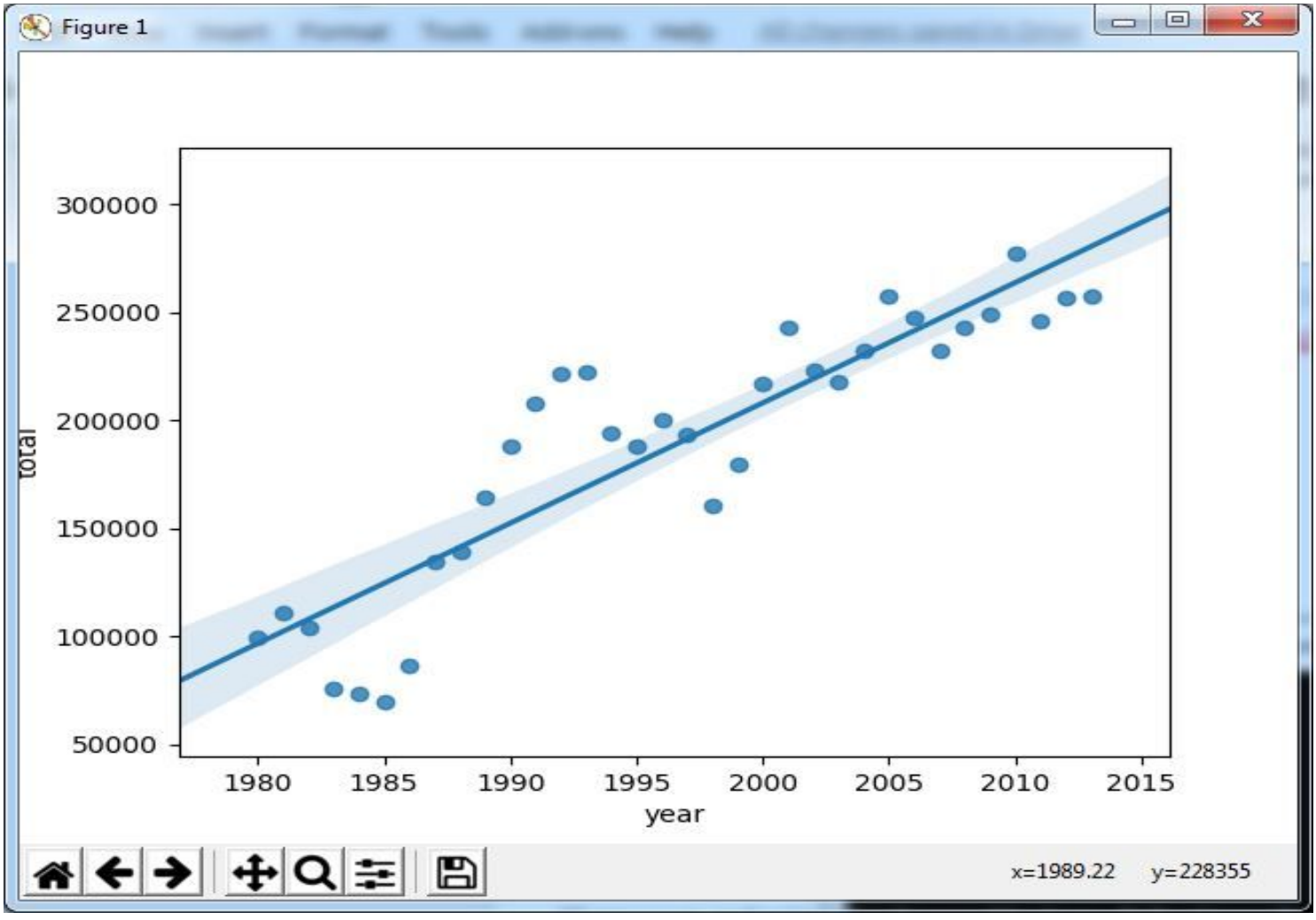


8. Regression Plot (Total immigrant population to Canada from 1980 to 2013):

Python Code:

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
df_can = pd.read_excel(
        'Canada.xlsx',
        sheet_name='Canada by Citizenship',
        skiprows=range(20),
        skipfooter=2
)
years=list(map(int,range(1980,2014)))
# we can use the sum() method to get the total population per year
df_tot = pd.DataFrame(df_can[years].sum())
# reset the index to put in back in as a column in the df_tot dataframe
df_tot.reset_index(inplace = True)
# rename columns
df_tot.columns = ['year', 'total']
a = sns.regplot(x='year',y='total',data=df_tot)
plt.show()
```

Output:



9. Canada Map:

Python Code:

```
import folium

world_map=folium.Map(
    location=[56.130,-106.35],
    zoom_start=4,
    tiles='Stamen Terrain'
)

world_map
```

Output:

10. World Map:

Python Code:

```
import folium

world_map=folium.Map(
        tiles="Stamen Terrain"
)

world_map
```

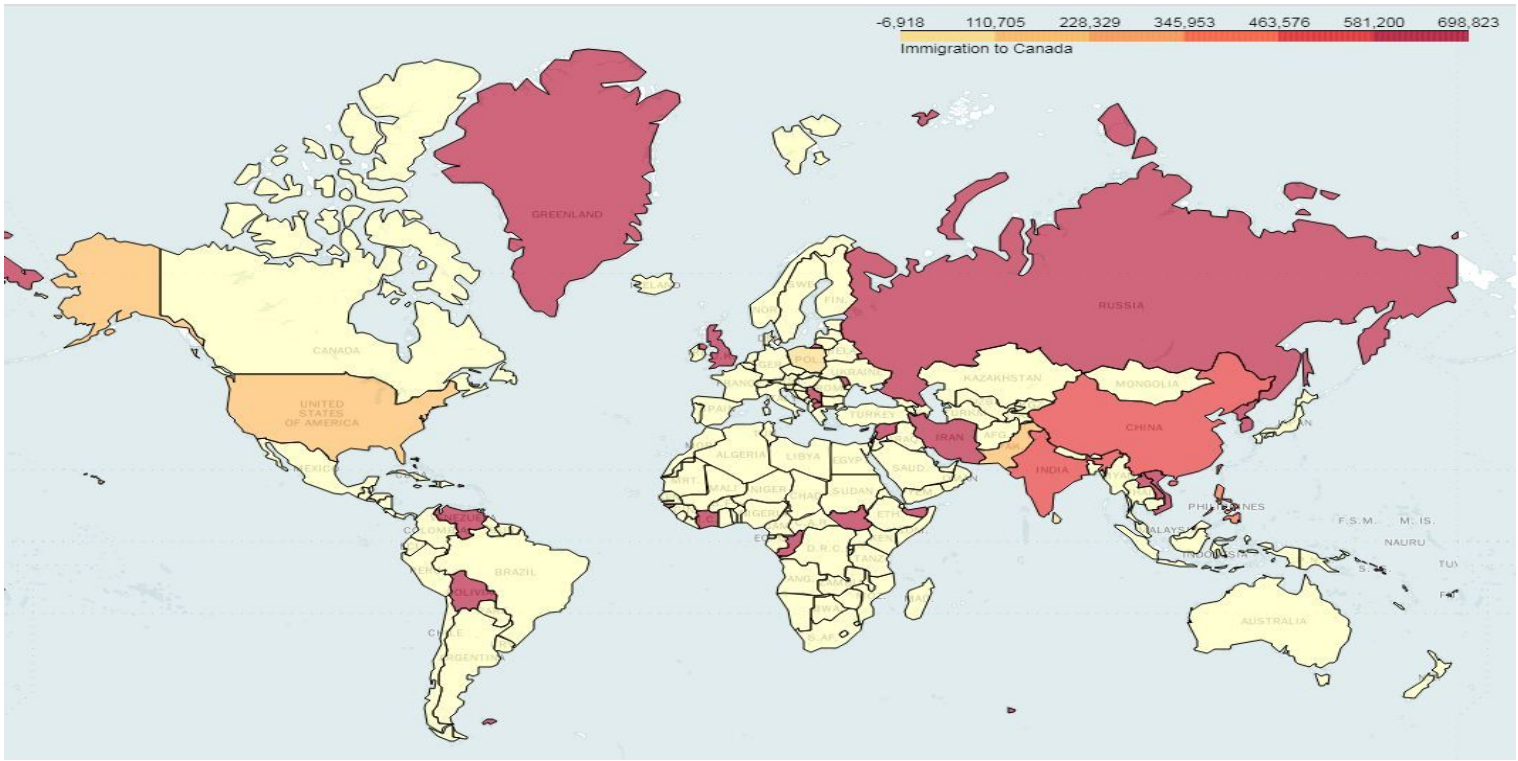
Output:

11. Choropleth Map (Total immigrant population to Canada from 1980 to 2013):

Python Code:

```
import folium
world_map=folium.Map(
    zoom_start=4,
    tiles='mapbox Bright'
)
world_geo='world_countries.json'
import pandas as pd
df_can=pd.read_excel(
            'Canada.xlsx',
            sheet_name='Canada by Citizenship',
            skiprows=range(20),
            skipfooter=2
)
df=df_can[['OdName','Total']]
world_map.choropleth(
    geo_data=world_geo,
    data=df,
    columns=['OdName','Total'],
    key_on='feature.properties.name',
    fill_color='YlOrRd',
    legend_name='Immigration to Canada'
)
world_map.save('chloro.html')
```

Output:

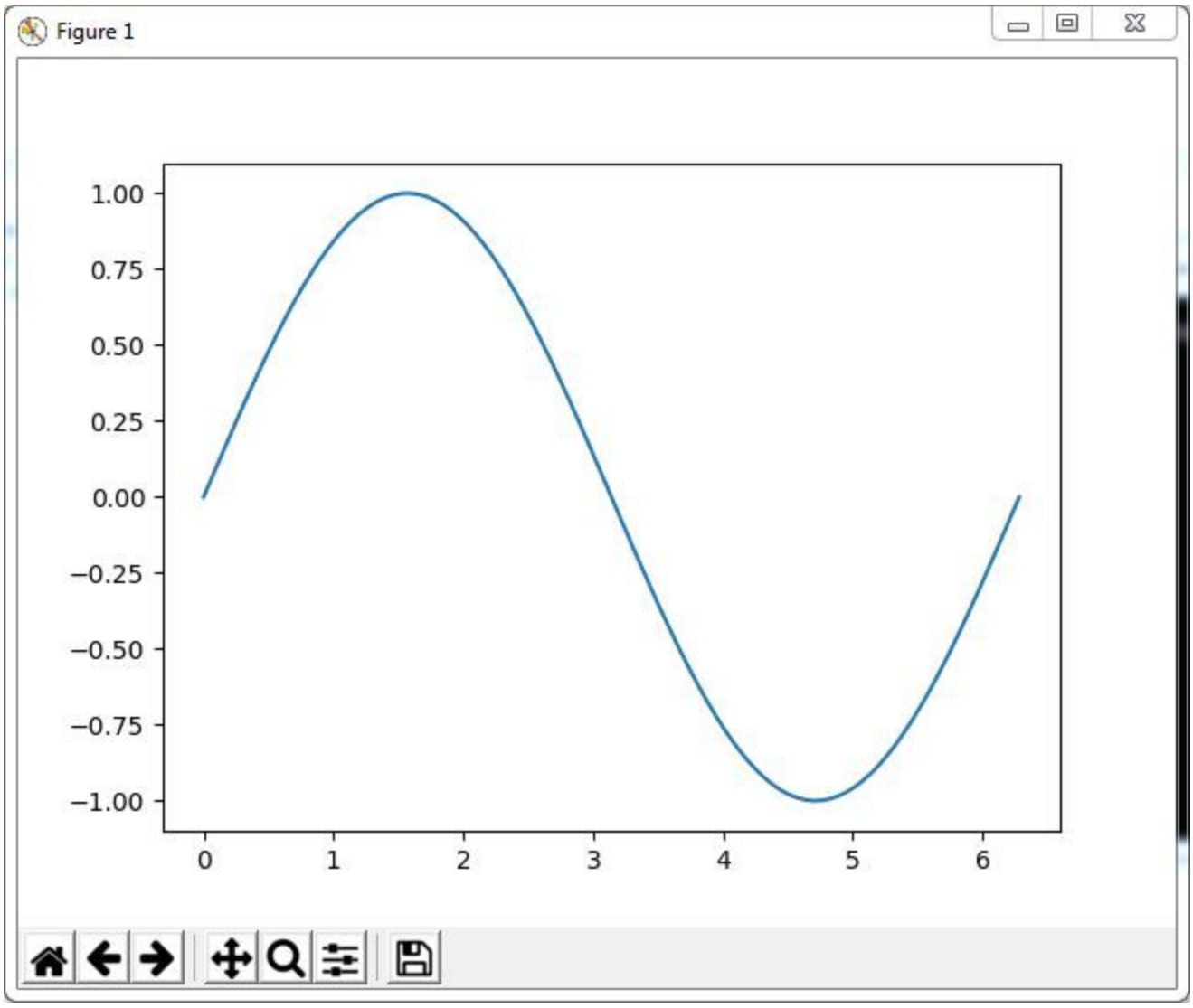


12. Sine Graph:

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
"""a=np.array([2,5])
b=np.array([6,7])
mean_a=np.mean(a)"""

"""a=np.array([0,np.pi/2,np.pi])
b=np.sin(a)
print(a)"""

a=np.linspace(0,2*np.pi,100)
b=np.sin(a)
plt.plot(a,b)
plt.show()
```

Output:



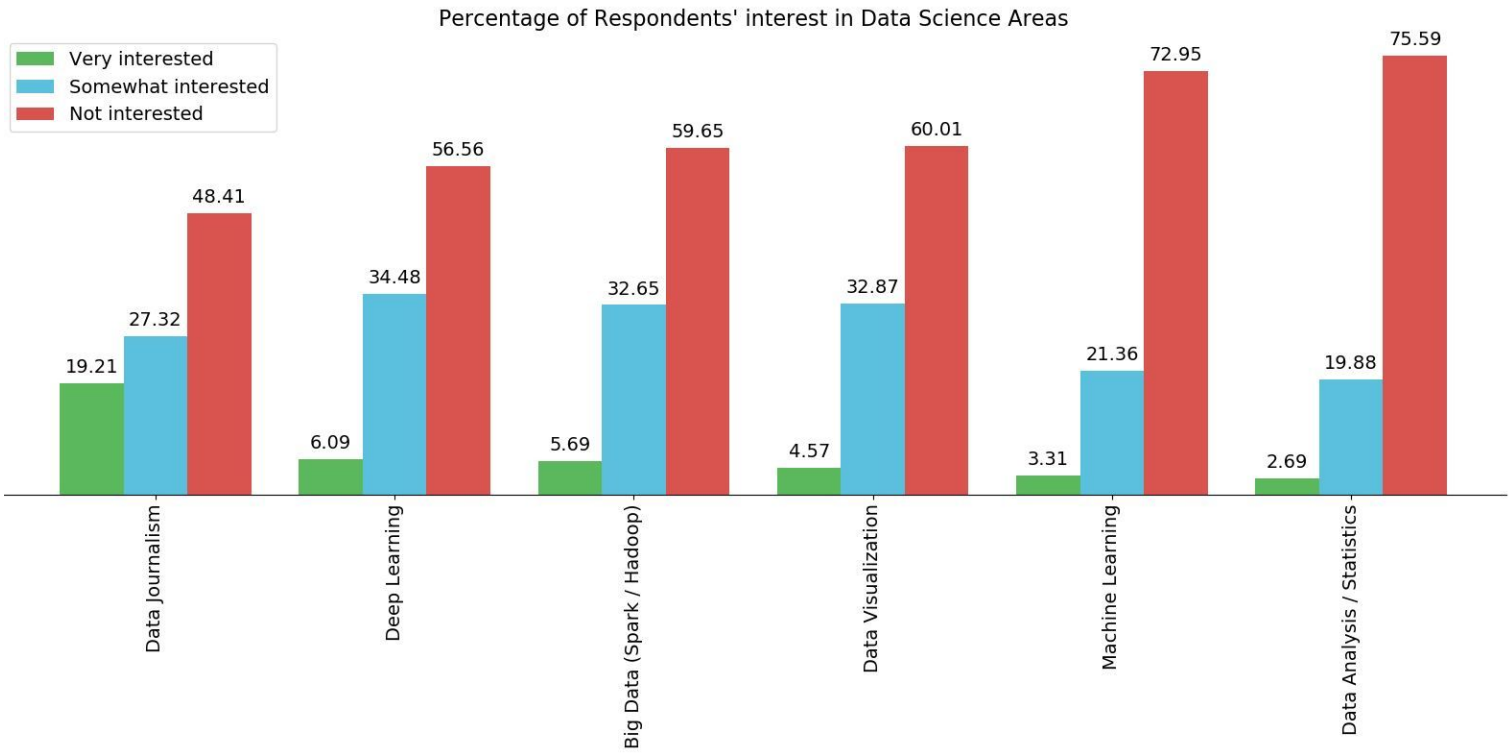
13. Bar Chart (Percentage of Respondents' interest in Data Science Areas):

Python Code:

```
import pandas as pd
import matplotlib.pyplot as plt

org = pd.read_csv('Data_Science_Topics_Survey.csv')

a={
        'Not interested':pd.Series([0,0,0,0,0,0],index=['Big Data (Spark / Hadoop)','Data Analysis / Statistics','Data Journalism','Data Visualization','Deep Learning','Machine Learning']),
        'Somewhat interested':pd.Series([0,0,0,0,0,0],index=['Big Data (Spark / Hadoop)','Data Analysis / Statistics','Data Journalism','Data Visualization','Deep Learning','Machine Learning']),
        'Very interested':pd.Series([0,0,0,0,0,0],index=['Big Data (Spark / Hadoop)','Data Analysis / Statistics','Data Journalism','Data Visualization','Deep Learning','Machine Learning'])
}

df = org["What's your level of interest for the following areas of Data Science? [Big Data (Spark / Hadoop)]"].value_counts()
a['Not interested'][0]=df[0]
a['Somewhat interested'][0]=df[1]
a['Very interested'][0]=df[2]

df = org["What's your level of interest for the following areas of Data Science? [Data Analysis / Statistics]"].value_counts()
a['Not interested'][1]=df[0]
a['Somewhat interested'][1]=df[1]
a['Very interested'][1]=df[2]

df = org["What's your level of interest for the following areas of Data Science? [Data Journalism]"].value_counts()
a['Not interested'][2]=df[0]
a['Somewhat interested'][2]=df[1]
a['Very interested'][2]=df[2]

df = org["What's your level of interest for the following areas of Data Science? [Data Visualization]"].value_counts()
a['Not interested'][3]=df[0]
a['Somewhat interested'][3]=df[1]
a['Very interested'][3]=df[2]

df = org["What's your level of interest for the following areas of Data Science? [Deep Learning]"].value_counts()
a['Not interested'][4]=df[0]
a['Somewhat interested'][4]=df[1]
a['Very interested'][4]=df[2]

df = org["What's your level of interest for the following areas of Data Science? [Machine Learning]"].value_counts()
a['Not interested'][5]=df[0]
a['Somewhat interested'][5]=df[1]
```

```
a['Very interested'][5]=df[2]

df_final = pd.DataFrame(a)

df_final = df_final[['Very interested','Somewhat interested','Not interested']]

df_final.sort_values(['Very interested'],ascending=False,axis=0,inplace=True)

for i in range(0,6):
    for j in range(0,3):
        df_final.ix[i,j]=round(((df_final.ix[i,j]*100)/2233),2)

col=['#5cb85c','#5bc0de','#d9534f']

ax = df_final.plot(kind='bar',figsize=(20,8),width=0.8,color=col)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.legend(fontsize=14)

def autolabel(rects, ax):
    for rect in rects:
        x = rect.get_x() + rect.get_width()/2.
        y = rect.get_height()
        ax.annotate("{}".format(y), (x,y), xytext=(0,5), textcoords="offset points",
                    ha='center', va='bottom',fontsize=14)

autolabel(ax.patches,ax)

ax.yaxis.set_visible(False)
plt.xticks(fontsize=14)

plt.title("Percentage of Respondents' interest in Data Science Areas",fontsize=16)
plt.tight_layout()
plt.show()
```

Output:

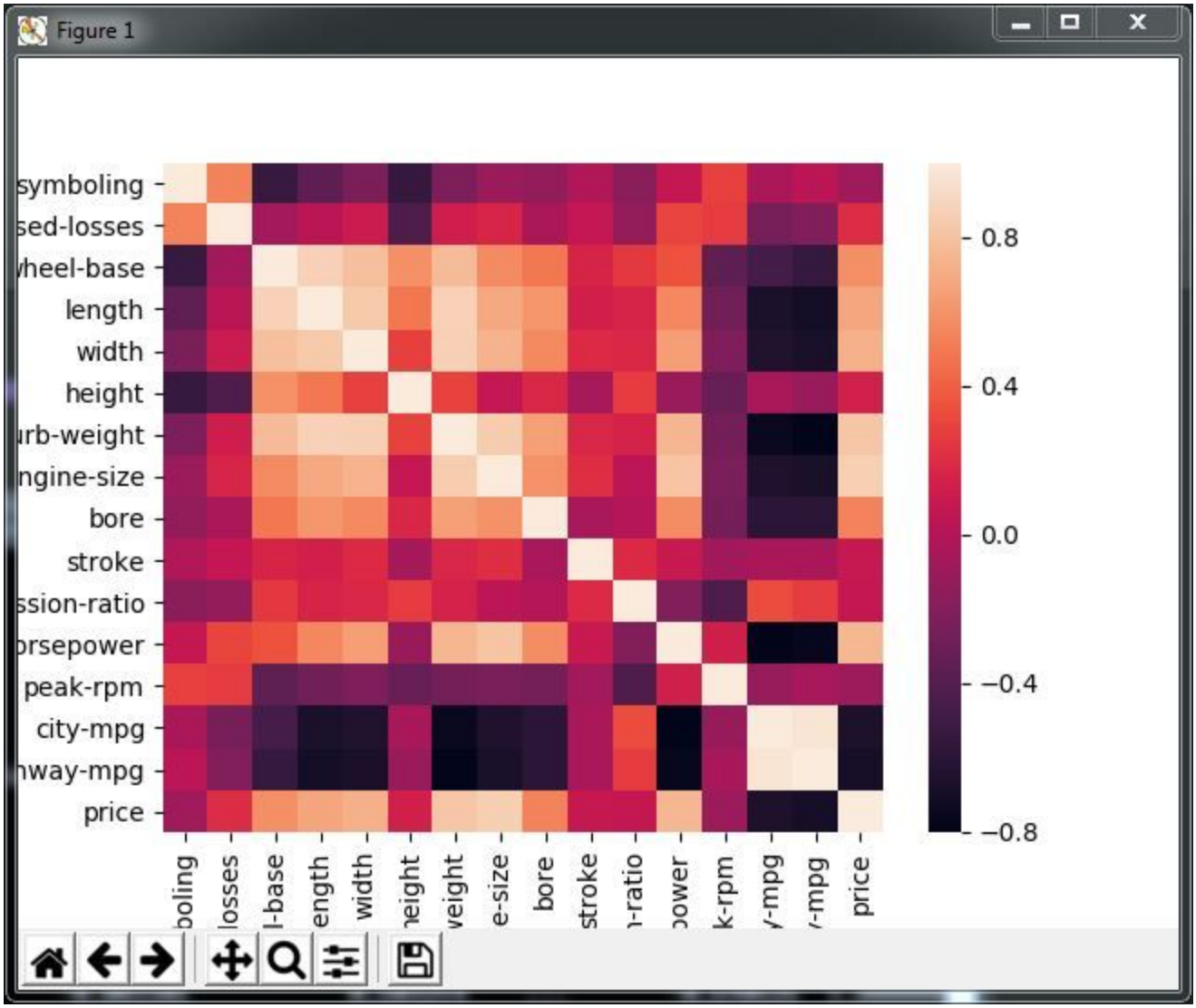


14. Heatmap (Pearson's Coefficient):

Python Code:

```
import pandas as pd
import scipy.stats as stats
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df=pd.read_excel('automobile.xlsx')

mean=df['horsepower'].mean()
df['horsepower'].replace(np.nan,mean,inplace=True)

mean=df['price'].mean()
df['price'].replace(np.nan,mean,inplace=True)

pearson_coeff,p_value=stats.pearsonr(df['horsepower'],df['price'])

print(pearson_coeff)
print(p_value)
print()

print(df.corr(method='pearson'))

ax=sns.heatmap(df.corr(method='pearson'))
plt.show()
```

Output:



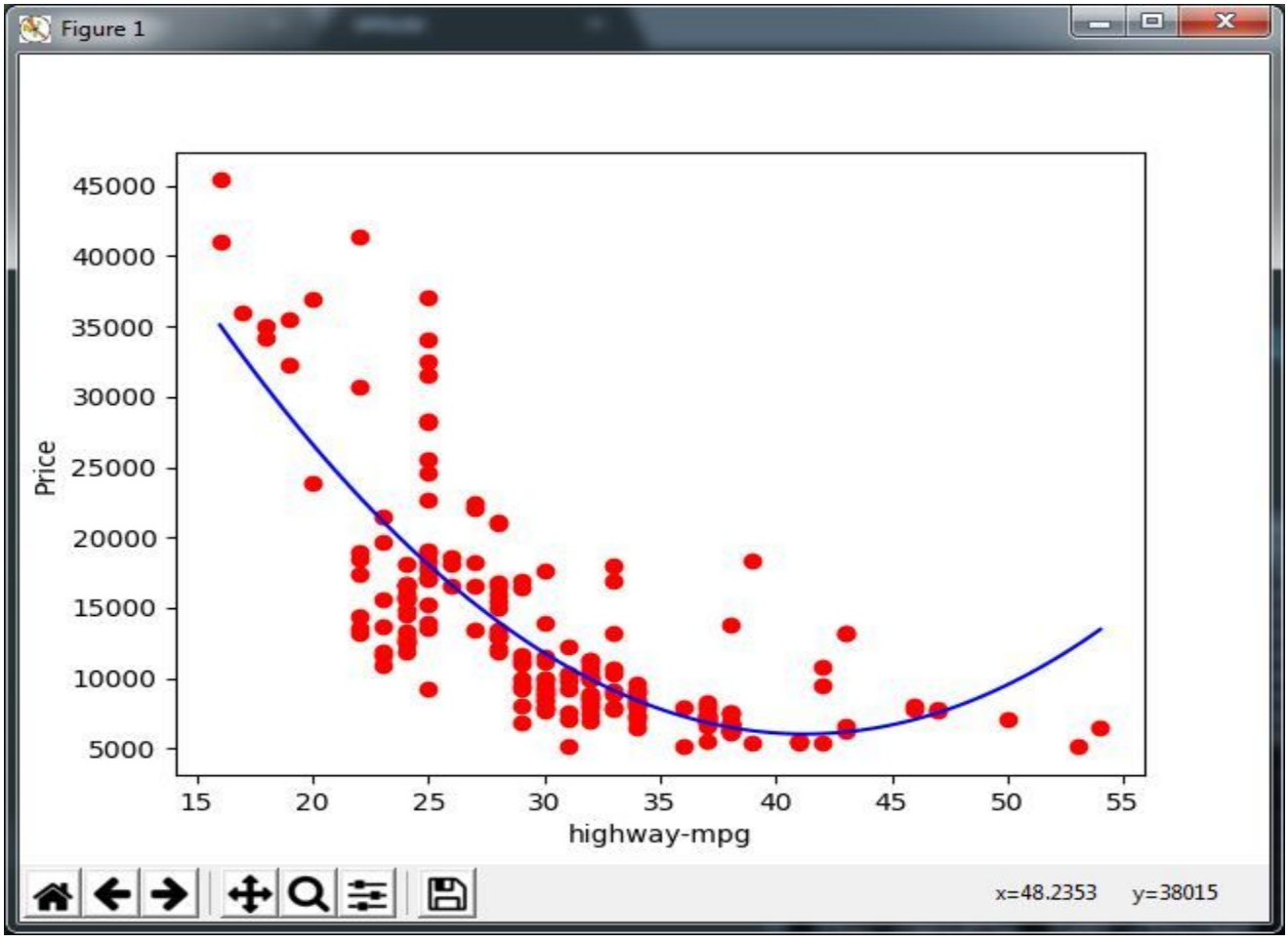
15. Polynomial Plot (Price is Quadratic function of Highway-mpg):

Python Code:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

df=pd.read_excel('automobile.xlsx')

mean=df['highway-mpg'].mean()
df['highway-mpg'].replace(np.nan,mean,inplace=True)

mean=df['engine-size'].mean()
df['engine-size'].replace(np.nan,mean,inplace=True)
```

```
mean=df['curb-weight'].mean()
df['curb-weight'].replace(np.nan,mean,inplace=True)

mean=df['horsepower'].mean()
df['horsepower'].replace(np.nan,mean,inplace=True)

mean=df['price'].mean()
df['price'].replace(np.nan,mean,inplace=True)

z = df[['highway-mpg']]
Input=[('scale',StandardScaler()),('polynomial',PolynomialFeatures(degree=2)),('model',LinearRegression())]
pipe=Pipeline(Input)
pipe.fit(z,df['price'])

plt.scatter(z,df['price'],color='red')
z=np.linspace(df['highway-mpg'].min(),df['highway-mpg'].max(),100)
y = np.reshape(z, (-1, 1))
ypipe=pipe.predict(y)
print(ypipe)
plt.plot(z,ypipe,color='blue')
plt.xlabel("highway-mpg")
plt.ylabel("Price")
plt.show()
```

Output:

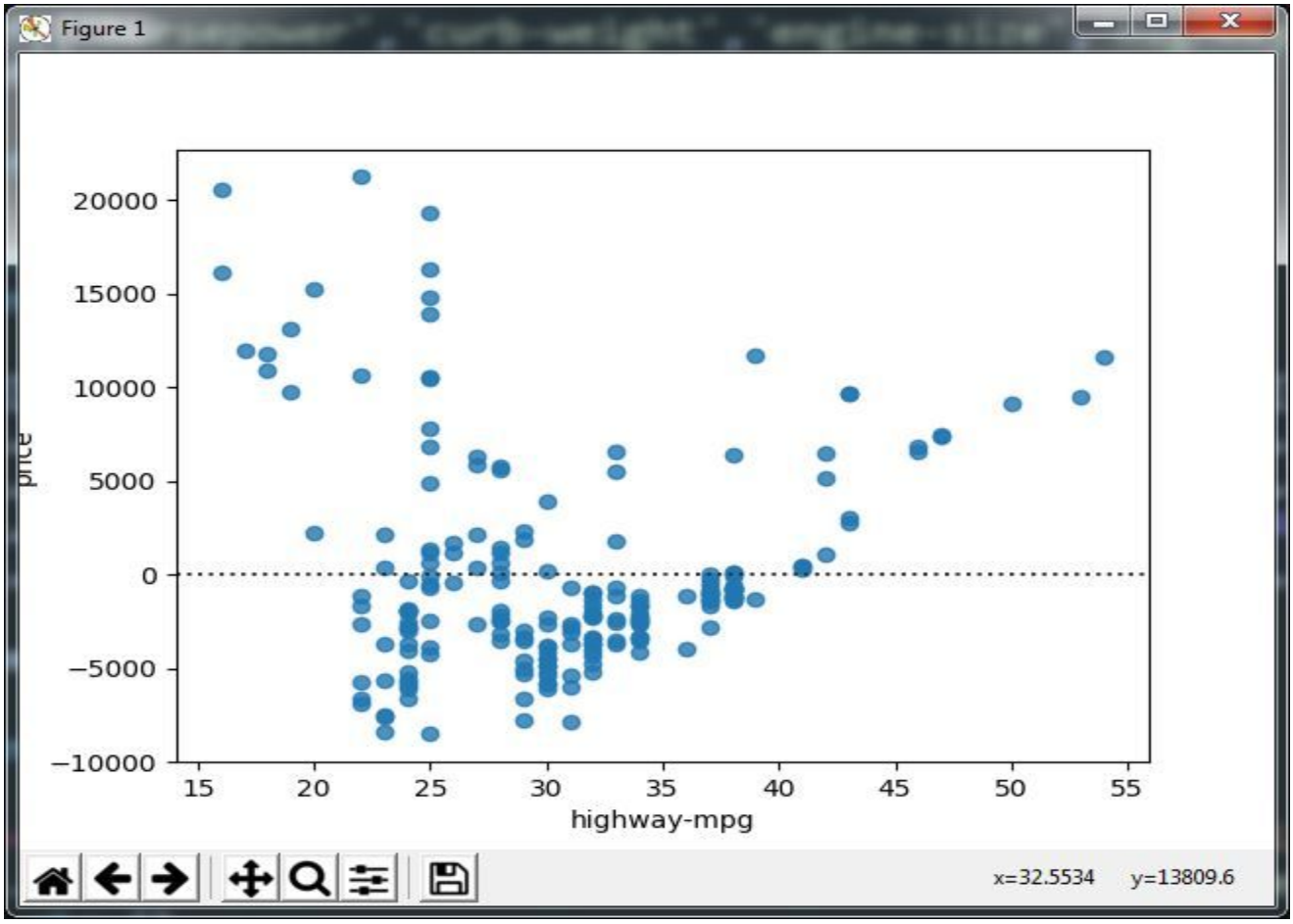


16. Residual and Distribution Plots (Used for Model Evaluation):

Python Code:

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

df=pd.read_excel('automobile.xlsx')

lm=LinearRegression()

mean=df['highway-mpg'].mean()
df['highway-mpg'].replace(np.nan,mean,inplace=True)

mean=df['horsepower'].mean()
df['horsepower'].replace(np.nan,mean,inplace=True)

mean=df['curb-weight'].mean()
df['curb-weight'].replace(np.nan,mean,inplace=True)

mean=df['engine-size'].mean()
df['engine-size'].replace(np.nan,mean,inplace=True)

mean=df['price'].mean()
df['price'].replace(np.nan,mean,inplace=True)

sns.residplot(df['highway-mpg'],df['price'])
plt.show()

x=df[['horsepower','curb-weight','engine-size','highway-mpg']]
y=df['price']

lm.fit(x,y)

print(lm.intercept_)
print(lm.coef_)

Y=lm.predict(x)

print(Y)

ax1=sns.distplot(df['price'],hist=False,color='r',label='Actual Value')
sns.distplot(Y,hist=False,color='b',label='Fitted Values',ax=ax1)
plt.ylabel("Properties of Car")
plt.show()
```

Residual Plot Output:



Distribution Plot Output: