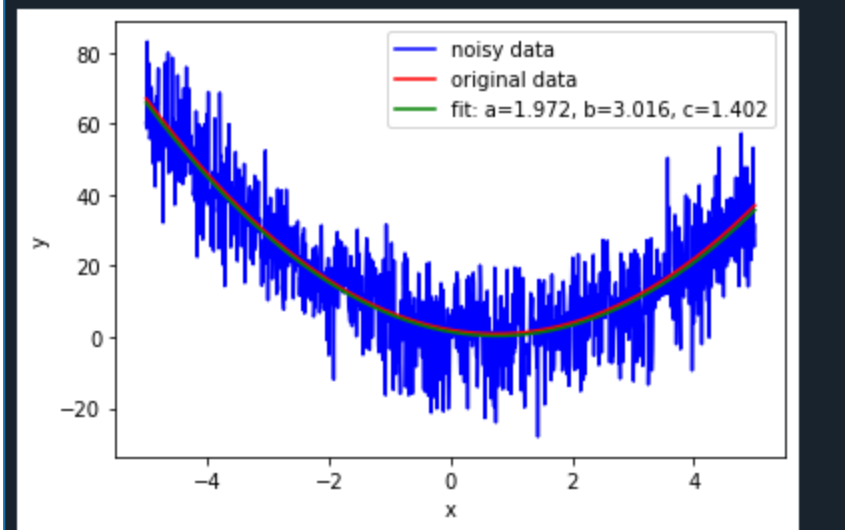
Solutions to Assignment 1 by Aether Zhou
This file is
https://docs.google.com/document/d/1j81eGkkfY_t5SSqg7SJBaleb2ggcb46y4CO0z4eBssI/edit?usp=sharing
Python filename is assignment_one_aether_zhou.py

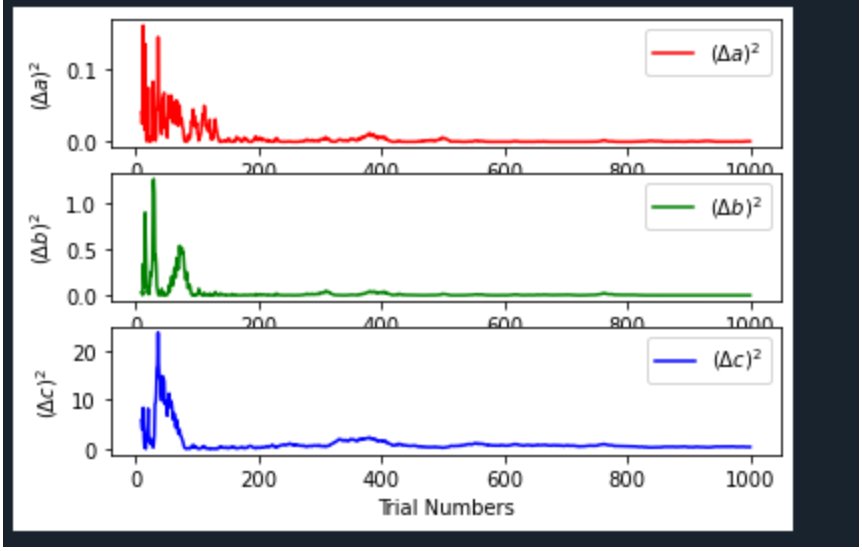
Plot with noise:

```
In [96]: plotMyPolynomialWithNoiseAndCurveFit()
[1.97180741 3.01553293 1.40150913]
```



Plot of square of the fitting error.

```
In [97]: plotMyFittingError()
```



Full codes:

```
# -*- coding: utf-8 -*-
"""
Created on Wed Oct 14 15:11:55 2020

@author: Aether Zhou
PHYS CS 15A
Week 2
"""
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit

def myPolynomial(x):
    return 2*x**2-3*x+2

def plotMyPolynomial():
    xValue = np.arange(-5.0, 5.0, 0.01)
    yValue = myPolynomial(xValue)

    fig, ax = plt.subplots()
    ax.plot(xValue, yValue)

    ax.set(xlabel='x', ylabel='f(x)', title='f(x)=2*x^2-3*x+2')
    ax.grid()

    plt.show()

def plotMyPolynomialWithNoise():
    xValue = np.arange(-5.0, 5.0, 0.01)
    yValue = myPolynomial(xValue)
    yNoise = np.random.normal(loc=myPolynomial(xValue), scale=10.0,
size=None)

    fig, ax = plt.subplots()

    ax.plot(xValue, yNoise)
    ax.plot(xValue, yValue)

    ax.set(xlabel='x', ylabel='f(x)', title='f(x)=2*x^2-3*x+2')
    ax.grid()

    plt.show()

def func(x, a, b, c):
```

```
    return a*x**2 -b*x+c

def plotMyPolynomialWithNoiseAndCurveFit():
    xdata = np.linspace(-5.0, 5.0, 1000)
    y = func(xdata, 2, 3, 2)
    np.random.seed(1729)

    yValue = myPolynomial(xdata)
    y_noise = 10 * np.random.normal(size=xdata.size)
    ydata = y + y_noise

    plt.figure()
    plt.plot(xdata, ydata, 'b-', label='noisy data')
    plt.plot(xdata, yValue, 'r-', label='original data')

    popt, pcov = curve_fit(func, xdata, ydata)
    print(popt)

    plt.plot(xdata, func(xdata, *popt), 'g-',
             label='fit: a=%5.3f, b=%5.3f, c=%5.3f' % tuple(popt))

    plt.xlabel('x')
    plt.ylabel('y')
    plt.legend()
    plt.show()

def plotMyFittingError():
    errorA=[]
    errorB=[]
    errorC=[]
    trialNum = range(10,1000)

    for i in trialNum:
        xdata = np.linspace(-5.0, 5.0, i)
        y = func(xdata, 2, 3, 2)
        np.random.seed(1729)

        y_noise = 10 * np.random.normal(size=xdata.size)
        ydata = y + y_noise
        popt, pcov = curve_fit(func, xdata, ydata)

        errorA.append((2-popt[0])**2)
        errorB.append((3-popt[1])**2)
        errorC.append((2-popt[2])**2)
```

```
plt.figure()
plt.subplot(311)
plt.plot(trialNum, errorA, 'r-', label=r'$(\Delta a)^2$')
plt.ylabel(r'$(\Delta a)^2$')
plt.legend()

plt.subplot(312)
plt.plot(trialNum, errorB, 'g-', label=r'$(\Delta b)^2$')
plt.ylabel(r'$(\Delta b)^2$')
plt.legend()

plt.subplot(313)
plt.plot(trialNum, errorC, 'b-', label=r'$(\Delta c)^2$')
plt.xlabel('Trial Numbers')
plt.ylabel(r'$(\Delta c)^2$')
plt.legend()

plt.show()
```