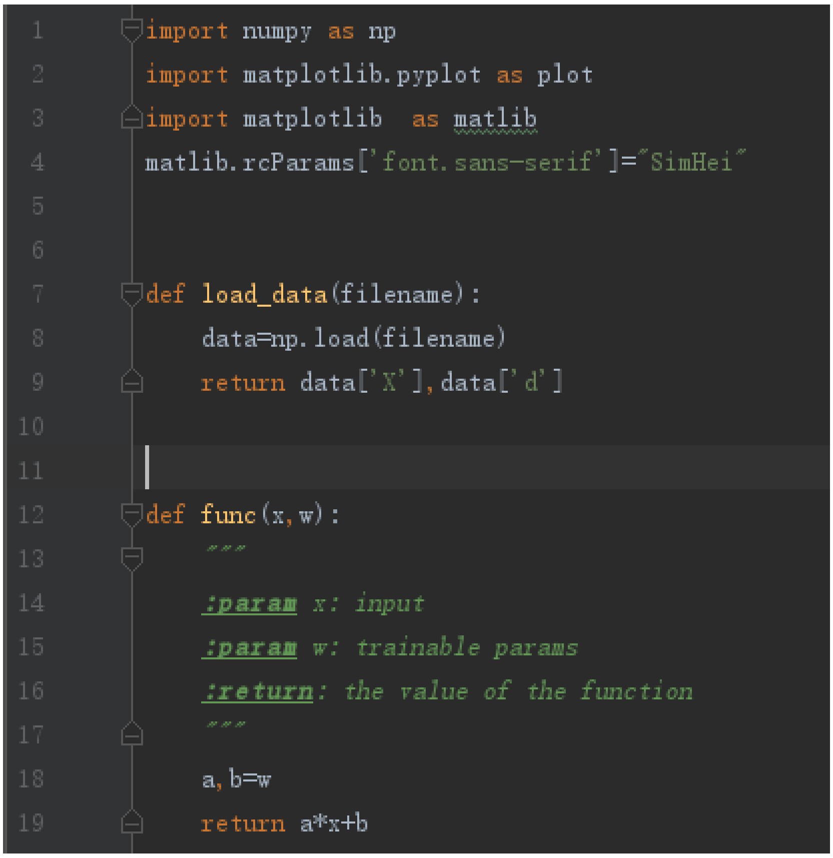
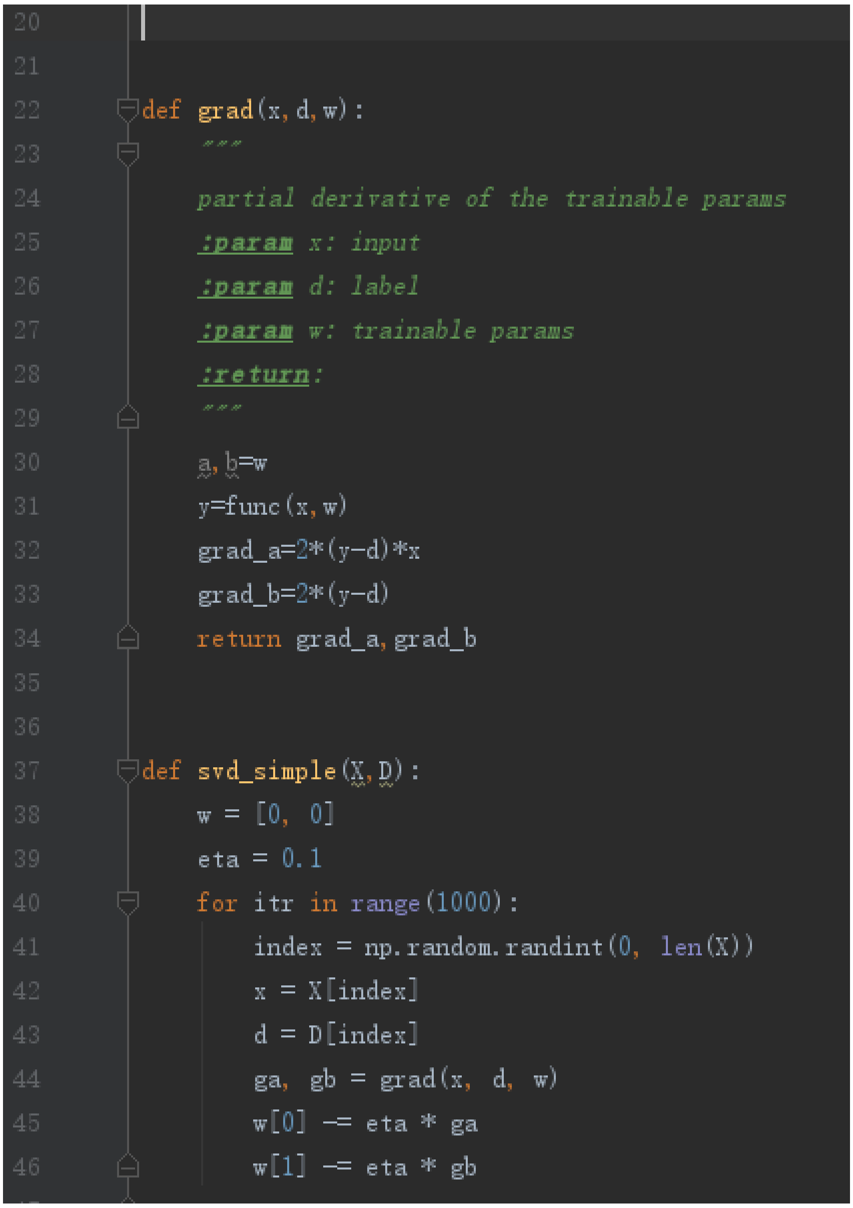
程序代码：

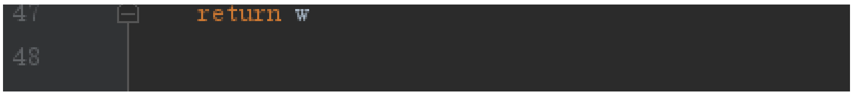
```
使用python库说明：
     python 3.7.0
     numpy  1.16.2
     matplotlib 3.0.0
```



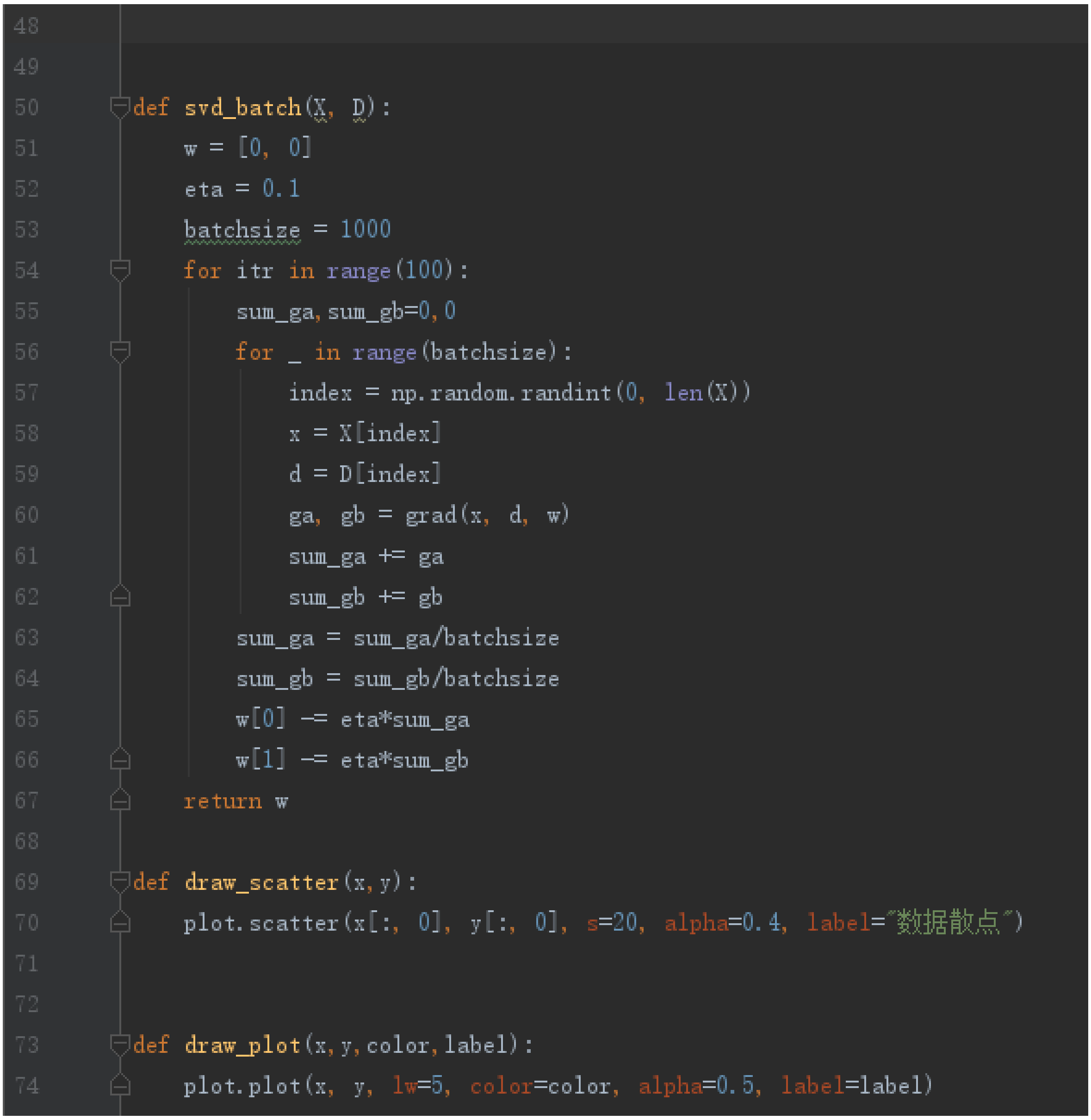
```python
import numpy as np
import matplotlib.pyplot as plot
import matplotlib  as matlib
matlib.rcParams['font.sans-serif']="SimHei"


def load_data(filename):
    data=np.load(filename)
    return data['X'],data['d']


def func(x,w):
    """

    :param x: input
    :param w: trainable params
    :return: the value of the function
    """
    a,b=w
    return a*x+b
```



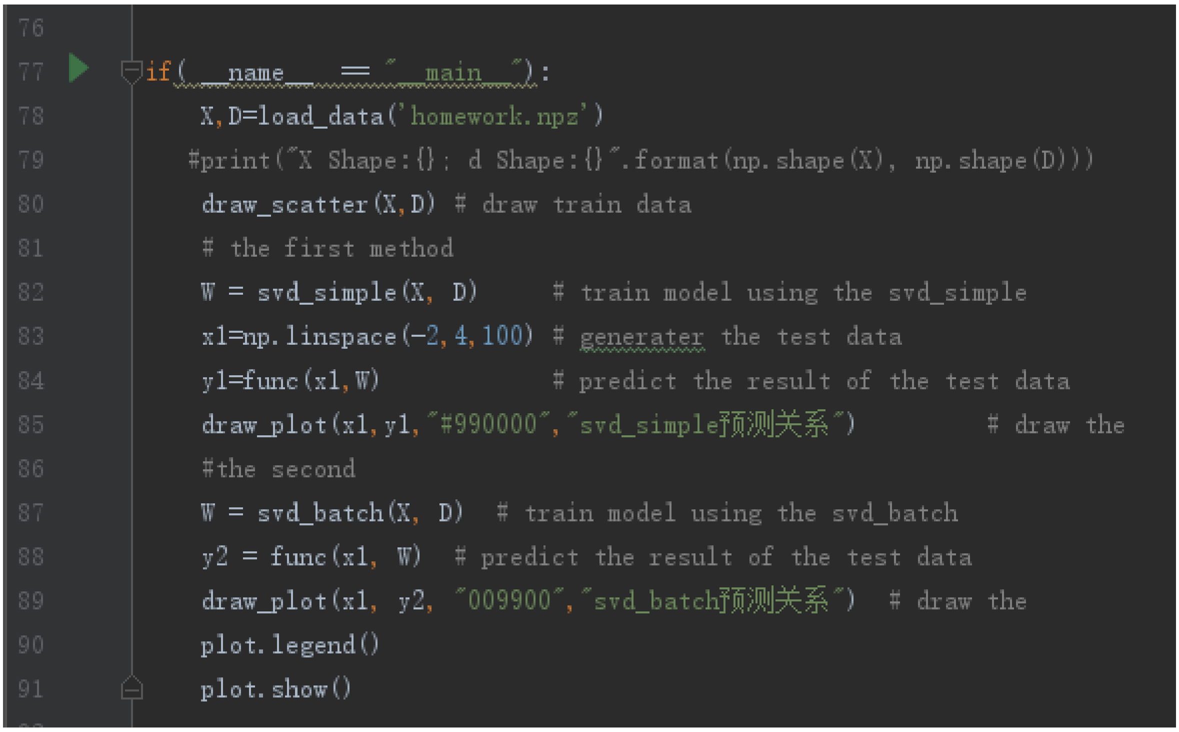
```python


def grad(x,d,w):
    """
    partial derivative of the trainable params
    :param x: input
    :param d: label
    :param w: trainable params
    :return:
    """
    a,b=w
    y=func(x,w)
    grad_a=2*(y-d)*x
    grad_b=2*(y-d)
    return grad_a,grad_b


def svd_simple(X,D):
    w = [0, 0]
    eta = 0.1
    for itr in range(1000):
        index = np.random.randint(0, len(X))
        x = X[index]
        d = D[index]
        ga, gb = grad(x, d, w)
        w[0] -= eta * ga
        w[1] -= eta * gb
```



```
    return w

```

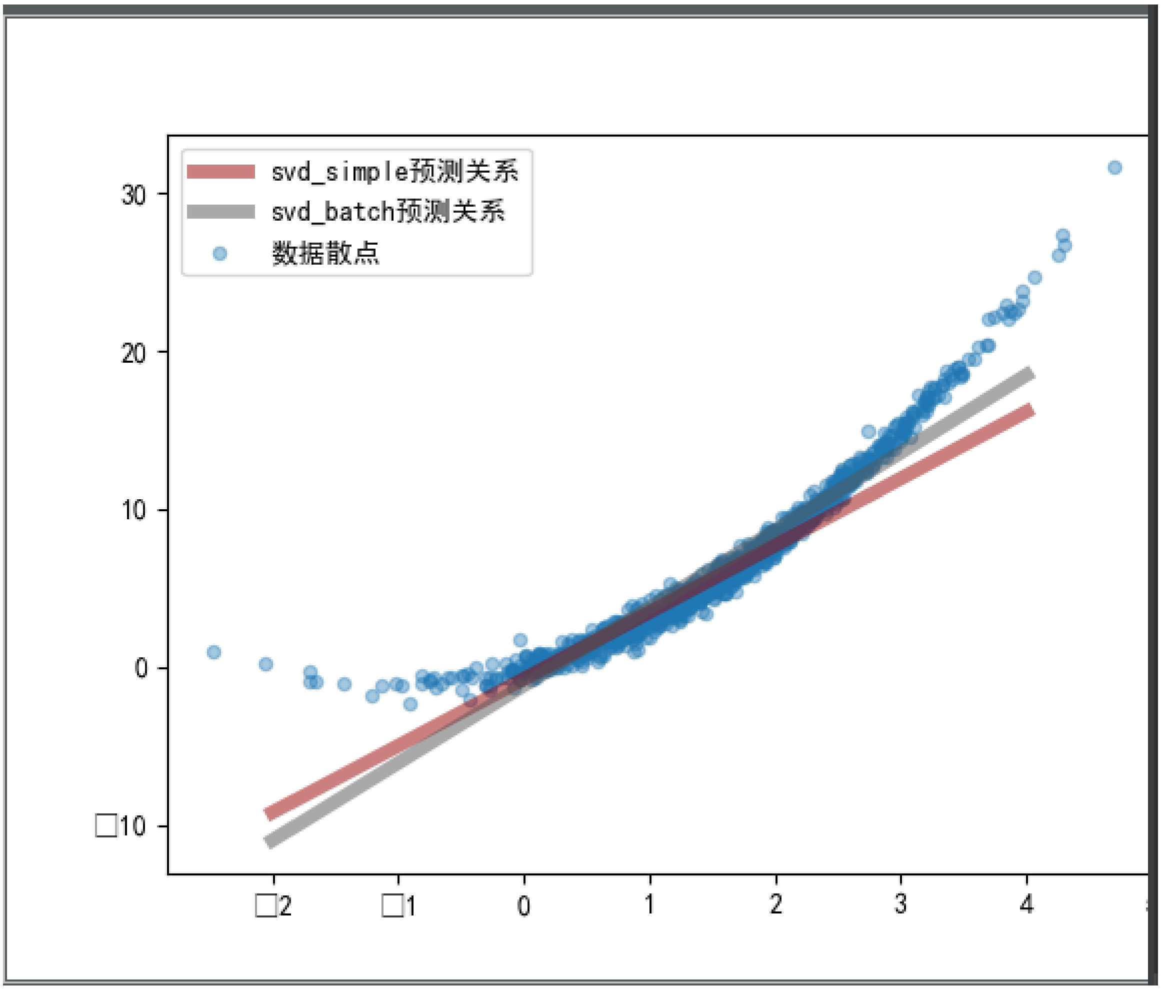


```


def svd_batch(X, D):
    w = [0, 0]
    eta = 0.1
    batchsize = 1000
    for itr in range(100):
        sum_ga,sum_gb=0,0
        for _ in range(batchsize):
            index = np.random.randint(0, len(X))
            x = X[index]
            d = D[index]
            ga, gb = grad(x, d, w)
            sum_ga += ga
            sum_gb += gb
        sum_ga = sum_ga/batchsize
        sum_gb = sum_gb/batchsize
        w[0] -= eta*sum_ga
        w[1] -= eta*sum_gb
    return w

def draw_scatter(x,y):
    plot.scatter(x[:, 0], y[:, 0], s=20, alpha=0.4, label="数据散点")


def draw_plot(x,y,color,label):
    plot.plot(x, y, lw=5, color=color, alpha=0.5, label=label)
```



```

if(__name__ == "__main__"):
    X,D=load_data('homework.npz')
   #print("X Shape:{}; d Shape:{}".format(np.shape(X), np.shape(D)))
    draw_scatter(X,D) # draw train data
    # the first method
    W = svd_simple(X, D)     # train model using the svd_simple
    x1=np.linspace(-2,4,100) # generater the test data
    y1=func(x1,W)            # predict the result of the test data
    draw_plot(x1,y1,"#990000","svd_simple预测关系")         # draw the
    #the second
    W = svd_batch(X, D)  # train model using the svd_batch
    y2 = func(x1, W)  # predict the result of the test data
    draw_plot(x1, y2, "009900","svd_batch预测关系")  # draw the
    plot.legend()
    plot.show()
```

执行结果：



是否有更好的建模方式？

> 应该是有的，根据数据散点图显示在`[-2,0]`区间的数据变化比较平缓，可以用分段数据训练，预测。

如何评价更好？

> 可以根据（预测值-样本值的平均值）**2来估算训练模型的拟合程度。