```
import numpy as np
from matplotlib import pyplot as plt
from q5_1 import *
import ipdb
import math
def calc_prob(x, lambda_, func):
    return np.exp(lambda_*func(x))

def MH_algo(lambda_, x, func, var):
    X = []
    x_ = x*1.0
    for i in range(500):
        while(1):
            y = np.random.normal(x_, var**0.5, 1)[0]
            if(y>=-5 and y<=5):
                break
        alpha = np.amin([calc_prob(y, lambda_, func)/calc_prob(x_, lambda_, func),1])
        if(math.isnan(alpha)):
            ipdb.set_trace()
        x_new = np.random.choice([y,x_], p=[alpha, 1-alpha])
        X.append(x_new*1.0)

    F = func(np.array(X))

    return X[np.argmax(F)], np.mean(X), np.mean(F)


def SA_algo(func, lambda_0, eta, var):

    x0 = np.random.uniform(-5, 5, 1)[0]
    x = x0
    lambda_ = lambda_0
    X = []
    F = []
    while (lambda_<=100):
        lambda_ = (1+eta)*lambda_
        x_new, x_mean, F_mean = MH_algo(lambda_, x, func, var)
        x = x_new*1.0
        X.append(x_mean*1.0)
        F.append(F_mean*1.0)

    return x, X, F

def plot_prop(data, prop_name):
    fig = plt.figure(figsize=(16,9))
    plt.plot(data)
    plt.xlabel("t")
    plt.ylabel(prop_name)
    plt.savefig("q5_3_{}.png".format(prop_name))
    plt.close()

def main():
    lambda_0 = 0.01
    eta = 0.1
    var = 0.1

    x_optim, X, F = SA_algo(f, lambda_0, eta, var)
    print(x_optim)
    plot_prop(X, "sampled_points_mean")
    plot_prop(F,"sampled_function_value_mean")

main()
```