```
import numpy as np
from matplotlib import pyplot as plt
import argparse


def f(x):
    val = x + np.sin(6*x)
    return val

def grad_f(x):

    grad = 1.0 + 6.0*np.cos(6*x)
    return grad

def grad_ascent(func, grad_func, x0, lr, num_steps):

    f_max = 5.464
    x_max = 4.478

    x_arr = np.zeros(shape=(num_steps,))
    f_arr = np.zeros(shape=(num_steps,))

    x = x0*1.0
    x_arr[0] = x*1.0
    f_arr[0] = (func(x))

    for i in range(1,num_steps):

        grad_f = grad_func(x)

        x = x + lr*grad_f
        x_arr[i] = x*1.0
        f_arr[i] = func(x)

    f_arr = f_max - f_arr
    x_arr = np.abs(x_arr - x_max)
    return f_arr, np.log(x_arr)


def plot_f(data_arr, lr_arr, ylabel=None, figname=None):
    fig = plt.figure(figsize=(16,9))

    for data, lr in zip(data_arr, lr_arr):
        plt.plot(data, label="step_size: {}".format(lr))
    plt.xlabel("step")
    if(ylabel):
        plt.ylabel(ylabel)
    plt.legend()
    plt.savefig("q51_{}.png".format(figname))
    plt.close()

def main():
    x0 = 0.0
    lrs = [0.01, 0.02, 0.05]
    f_arr = []
    x_arr = []
    num_steps = 100
    for lr in lrs:
        f_vals, xs = grad_ascent(f, grad_f, x0, lr, num_steps)
        f_arr.append(f_vals)
        x_arr.append(xs)

    plot_f(f_arr, lrs, ylabel="(f* - f(xt))", figname="f_vals")
    plot_f(x_arr, lrs, ylabel="log(|xt - x*|)", figname="x_vals")

if __name__ == "__main__":
    main()
```