例 3.11 求目标函数在给定点的最速下降方向

```
修改参数：f:目标函数
        X_0：给定点
"""

import numpy as np
from sympy import *
from sklearn.preprocessing import Normalizer


def steepest_descent_direction():
    # 设置变量
    x1 = symbols("x1")
    x2 = symbols("x2")

    # 求偏导
    difyL_x1 = diff(f, x1)
    difyL_x2 = diff(f, x2)

    # 求在给定点的最速下降方向
    difyL_x1 = difyL_x1.subs([(x1, X_0[0][0]), (x2, X_0[1][0])])
    difyL_x2 = difyL_x2.subs([(x1, X_0[0][0]), (x2, X_0[1][0])])
    P = np.array([[-difyL_x1],
                  [-difyL_x2]
                  ])
    print("f 在 X0 的最速下降方向:"'\n', P)

    # 计算该方向上的单位向量
    P_trans = np.transpose(P)
    scaler = Normalizer().fit(P_trans)
    P_scaler = scaler.transform(P_trans)
    e = np.transpose(P_scaler)

    # 新点
    X_1 = X_0 + e

    # 目标函数值
    f_X1 = f.subs([(x1, X_1[0][0]), (x2, X_1[1][0])])
    print("沿这个方向移动一个单位长度后新点的目标函数值为：", f_X1)


if __name__ == '__main__':
    x1 = symbols("x1")
    x2 = symbols("x2")
```

```
f = 3 * x1 ** 2 - 4 * x1 * x2 + x2 ** 2
X_0 = np.array([[0],
                [1]
                ])
steepest_descent_direction()
```