Congratulations! You passed!

TO PASS 80% or higher

Keep Learning

GRADE
100%

Lagrange multipliers

TOTAL POINTS 5

1. In this quiz we will consider Lagrange multipliers as a technique to find a minimum of a function subject to a constraint, i.e. solutions lying on a particular curve.

1 / 1 point

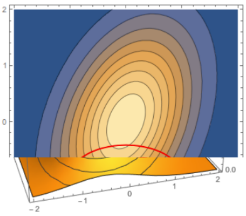
Let's consider the example of finding the minimum of the function,

$$f(\mathbf{x}) = \exp\left(-\frac{2x^2 + y^2 - xy}{2}\right)$$

along the curve (or, subject to the constraint),

$$g(\mathbf{x}) = x^2 + 3(y+1)^2 - 1 = 0\,.$$

The functions themselves are fairly simple, on a contour map they look as follows,



Do note, in this case, the function $f(\mathbf{x})$ does not have any minima itself, but along the curve, there are two minima (and two maxima).

A situation like this is where Lagrange multipliers come in. The observation is that the maxima and minima on the curve, will be found where the constraint is parallel to the contours of the function.

Since the gradient is perpendicular to the contours, the gradient of the function and the gradient of the constraint will also be parallel, that is,

$$\nabla f(\mathbf{x}) = \lambda \nabla g(\mathbf{x})$$

If we write this out in component form, this becomes,

Let's set up the system,

The function and two of the derivatives are defined for you. Set up the other two by replacing the question marks in the following code.

```
# First we define the functions,
def f (x, y) :
    return np.exp(-(2*x*x + y*y - x*y) / 2)

def g (x, y) :
    return x*x + 3*(y+1)**2 - 1

# Next their derivatives,
def dfdx (x, y) :
    return 1/2 * (-4*x + y) * f(x, y)

def dfdy (x, y) :
    return 1/2 * (x - 2*y) * f(x, y)

def dgdx (x, y) :
    return 2 * x

def dgdy (x, y) :
    return 3 * 2 * (y + 1)
```

Run

Reset

```
# (x0, y0, λ0) = (-1, -1, 0)
# x, y, λ = optimize.root(DL, [x0, y0, λ0]).x
# print("x = %g" % x)
# print("y = %g" % y)
# print("λ = %g" % λ)
# print("f(x, y) = %g" % f(x, y))

from scipy import optimize

def DL (xyλ) :
    [x, y, λ] = xyλ
    return np.array([
            dfdx(x, y) - λ * dgdx(x, y),
            dfdy(x, y) - λ * dgdy(x, y),
            - g(x, y)
        ])

(x0, y0, λ0) = (1, 0, 0)
x, y, λ = optimize.root(DL, [x0, y0, λ0]).x
print("x = %g" % x)
print("y = %g" % y)
print("λ = %g" % λ)
print("f(x, y) = %g" % f(x, y))
```

Run

Reset

```
x = 0.930942
y = -1.21083
```

Correct

This is one of the maxima or minima.

3. In the previous question, you gave the y coordinate of *any* of the stationary points. In this part, give the x coordinate of the *global minimum* of $f(x)$ on $g(x) = 0$.

1 / 1 point

Give your answer to 2 decimal places.

0.93

Correct

Eactly this is the global minimum, subject to the constraint.

5. Hopefully you've now built up a feeling for how Lagrange multipliers work. Let's test this out on a new function and constraint.

1 / 1 point

Calculate the minimum of

$$f(x, y) = -\exp(x - y^2 + xy)$$

on the constraint,

$$g(x, y) = \cosh(y) + x - 2 = 0$$

Use the code you've written in the previous questions to help you.

```
# Import libraries
import numpy as np
from scipy import optimize

# First we define the functions, YOU SHOULD IMPLEMENT THESE
def f (x, y) :
# Use the definition of DL from previously.
def DL (xyλ) :
    [x, y, λ] = xyλ
    return np.array([
            dfdx(x, y) - λ * dgdx(x, y),
            dfdy(x, y) - λ * dgdy(x, y),
            - g(x, y)
        ])

# To score on this question, the code above should set
# the variables x, y, λ, to the values which solve the
# Langrange multiplier problem.

# I.e. use the optimize.root method, as you did previously.
(x0, y0, λ0) = (0, 0, 0)
x, y, λ = optimize.root(DL, [x0, y0, λ0]).x

print("x = %g" % x)
print("y = %g" % y)
print("λ = %g" % λ)
print("f(x, y) = %g" % f(x, y))
```

Run

Reset

```
x = 0.957782
y = 0.289565
λ = -4.07789
f(x, y) = -3.16222
```

Correct

Well done. You've constucted and run your own Lagrange multiplyer solver for a new function.