ENGR 21
Computer Engineering Fundamentals

Instructor: Emad Masroor

Lecture 20
Thursday, Nov 13, 2025

Solving 1-D Optimization problems with Python

Find a minimum of the function

$$f(x) = x^2 - 4x + 7$$

Between [-1 , +5].

```
pip install scipy
```

get code from the Resources page !

```
from scipy.optimize import minimize_scalar
import numpy as np
from matplotlib import pyplot as plt

def test_f(x):
    return x**2 - 4*x + 7

solution = minimize_scalar(test_f,[-1,5])

print(f"Function minimized at x = {solution.x:.4f}")

x = np.linspace(-1,5)
plt.plot(x,test_f(x))
plt.grid()
plt.xlabel('y')
plt.ylabel('Shear stress')

plt.scatter(solution.x,test_f(solution.x),marker='o',color='black')
plt.show()
```

```
>>> solution
 message:
          Optimization terminated
successfully;
          The returned value
satisfies the termination criteria
          (using xtol = 1.48e-08 )
 success: True
     fun: 3.0
       x: 2.0
     nit: 5
    nfev: 8
```

2 Solving n-D Optimization problems with Python

Find a minimum of the function

$$f(x,y) = (x-1)^2 + (y+1)^2 + xy$$

Near the origin.

```
pip install scipy
```

```
from scipy.optimize import minimize
import numpy as np

def f(xx):
    x = xx[0]
    y = xx[1]
    return (x-1)**2 + (y+1)**2 + x*y

solution = minimize(f,np.array([0.,0.]),method='Powell')
# followed by more code to plot the result
```

get code from the Resources page !

Solving n-D Optimization problems with Python

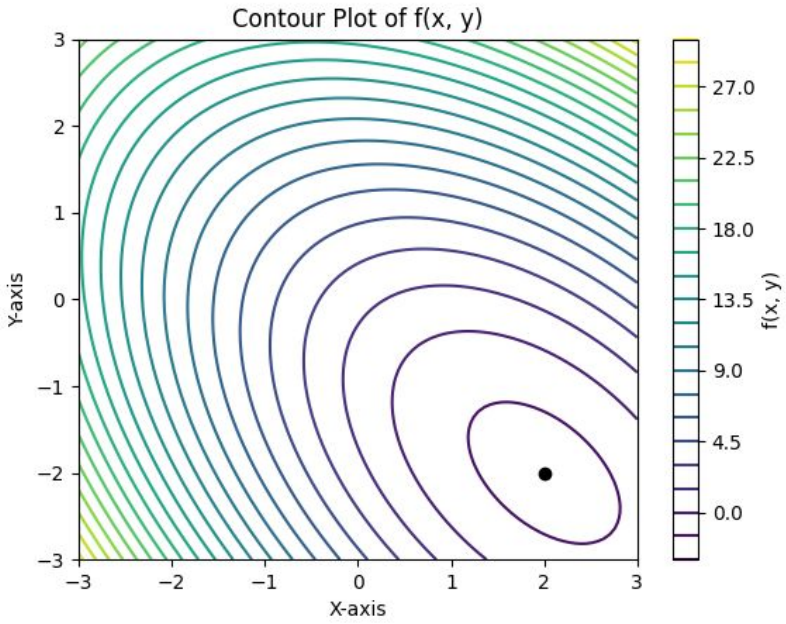
Find a minimum of the function

$$f(x, y) = (x - 1)^2 + (y + 1)^2 + xy$$

Near the origin.

```
pip install scipy
```

```
x = np.linspace(-3, 3, 100)
y = np.linspace(-3, 3, 100)
X, Y = np.meshgrid(x, y)

# Calculate the function values over the grid
Z = f([X,Y])

# Create a contour plot
contours = plt.contour(X, Y, Z, levels=20, cmap='viridis')

# Add labels and a color bar
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.colorbar(contours, label='f(x, y)')

# plot the minimum
plt.scatter(solution.x[0],solution.x[1],marker='o',color='black')

# Show the plot
plt.title('Contour Plot of f(x, y)')
plt.show()
```



get code from the Resources page !

3

Solving n-D Optimization problems with Python

with constraints!

Find a minimum of the function

$$f(x,y) = (x-1)^2 + (y+1)^2 + xy$$

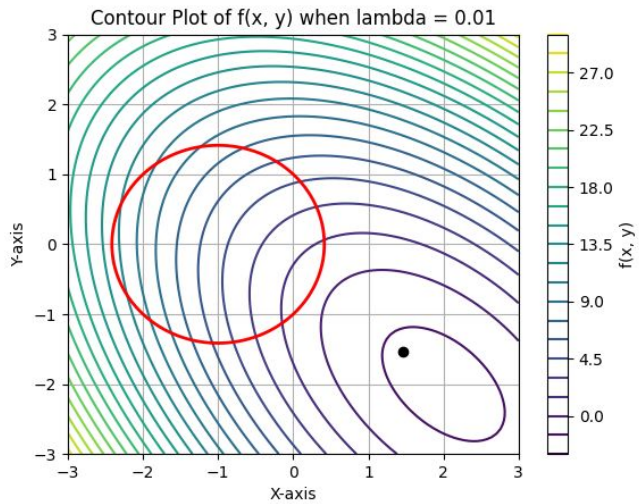
```
pip install scipy
```

Subject to the constraint that the minimum should fall on the circle

$$(x+1)^2 + y^2 - 2 = 0$$

Circle with radius √2 centered at (-1, 0).

```
# GLOBAL VARIABLE
LAMBDA = 0.01

def f(xx):
    x = xx[0]
    y = xx[1]
    return (x-1)**2 + (y+1)**2 + x*y

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

def f_star(x_vector):
    x = x_vector[0]
    y = x_vector[1]
    lam = LAMBDA      # parameter to change
    return f(x_vector) + lam*(constraint(x,y))**2

solution = minimize(f_star,np.array([0.,0.]),method='Powell')
```



Find a minimum of the function

$$f(x,y) = (x-1)^2 + (y+1)^2 + xy$$

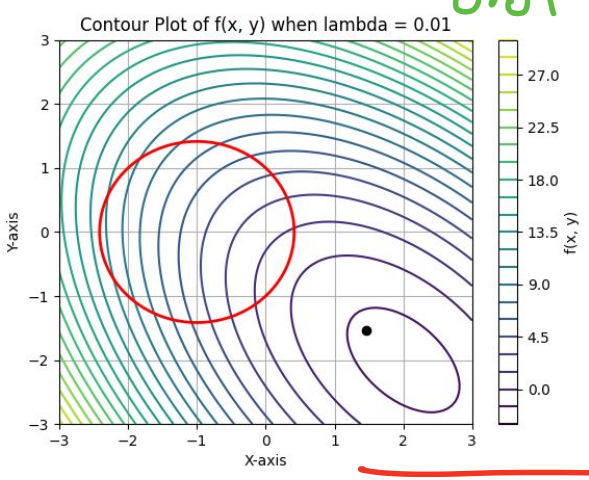
Subject to the constraint that the minimum should fall on the circle

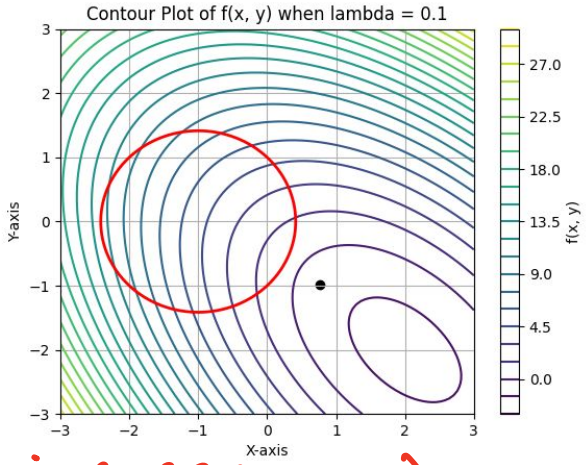
$$(x+1)^2 + y^2 - 2 = 0$$

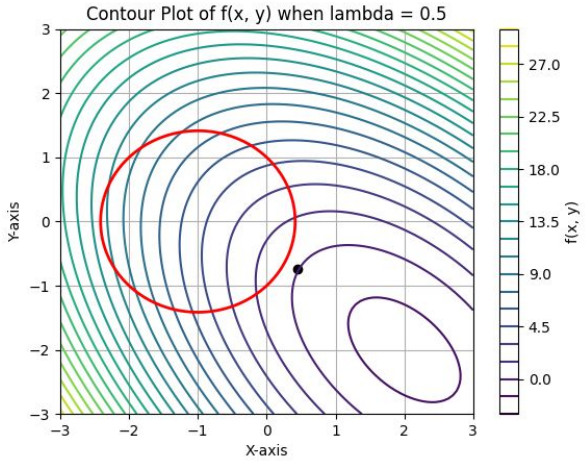
How does changing λ affect result?

small λ → increase λ

0.01

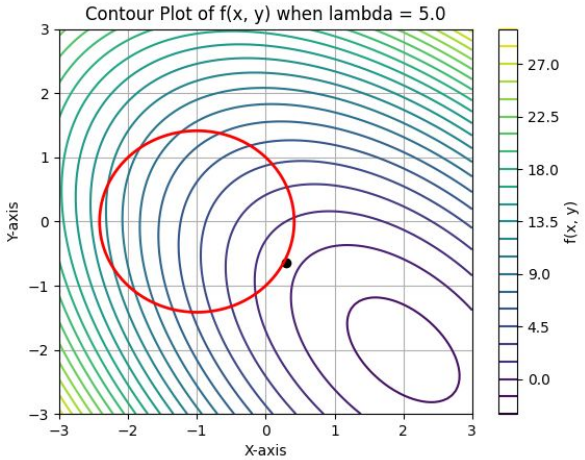


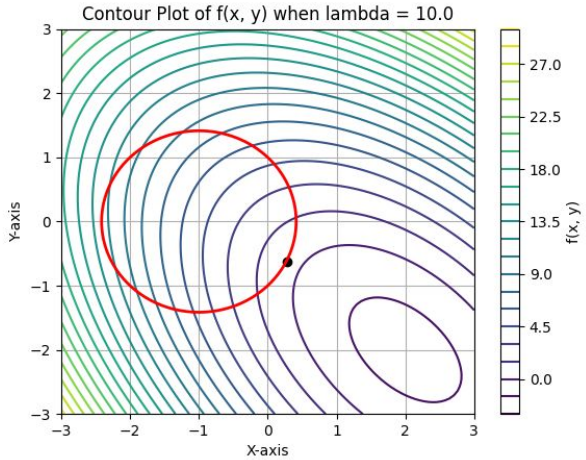


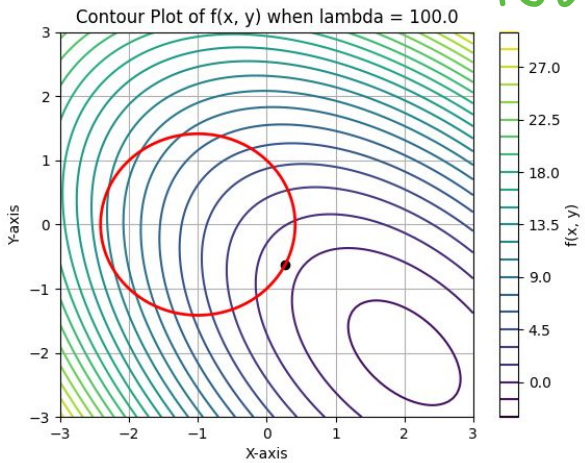


increase λ

100.0







Illustrating Naive n-d optimization

$$f([x,y]) = (x-1)^2 + (y+1)^2 + xy$$

Minimize $f(\boldsymbol{x_0} + \alpha \boldsymbol{v_1})$

$$f\left(\begin{bmatrix}-1\\2\end{bmatrix} + \alpha\begin{bmatrix}0\\1\end{bmatrix}\right)$$

$$f\left(\begin{bmatrix}-1+0\alpha\\2+1\alpha\end{bmatrix}\right)$$

$$= (-1-1)^2 + (2+\alpha+1)^2 + (-1+0)(2+\alpha)$$

$$= 4 + \alpha^2 + 6\alpha + 9 - 2 - \alpha$$

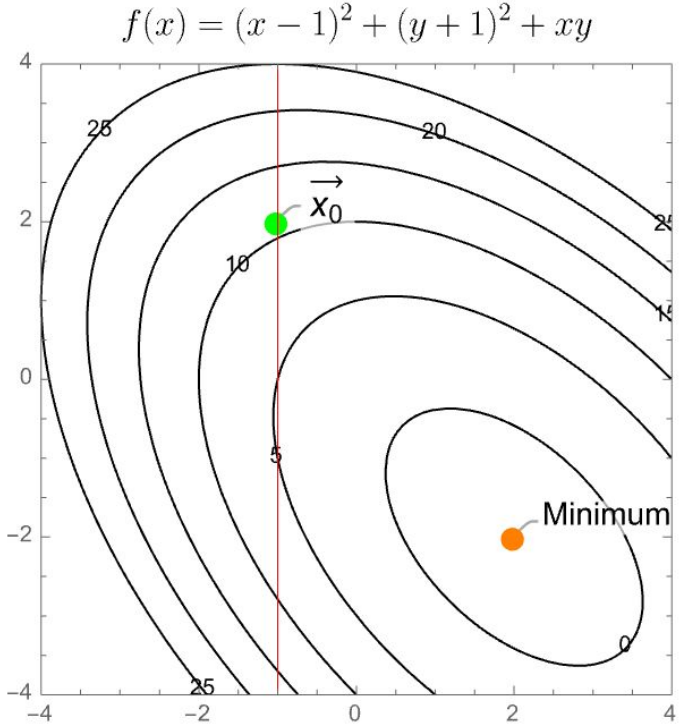
$$= \alpha^2 + 5\alpha + 11$$

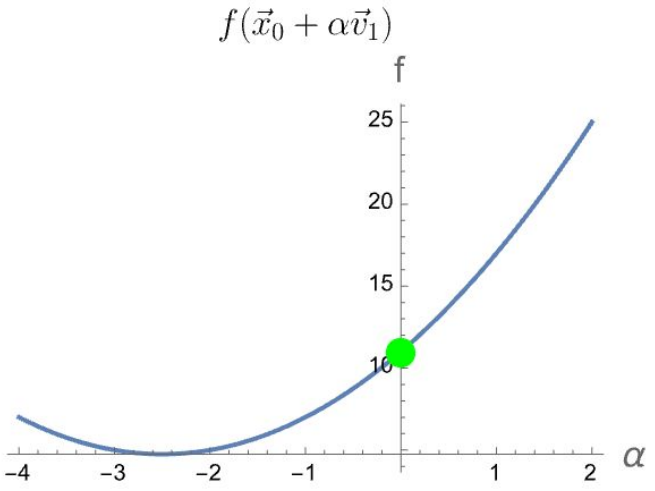
Minimize $\alpha^2 + 5\alpha + 11$

$$\implies \alpha_{\min} = -5/2$$

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \alpha_{\min}\boldsymbol{v}_1$$

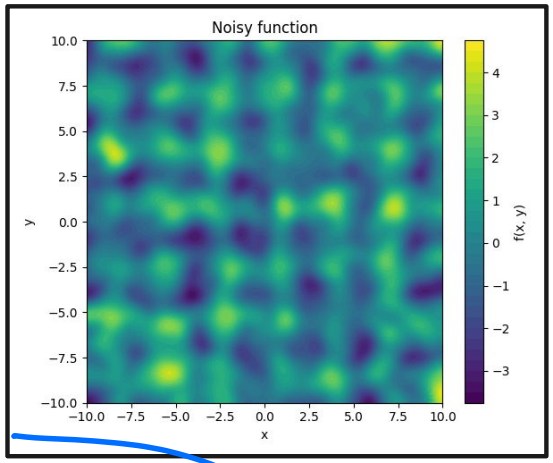
$$= \begin{bmatrix}-1\\2\end{bmatrix} - \frac{5}{2}\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}-1\\-0.5\end{bmatrix}$$





In-class task + HW 10 assignment: Python code for naive n-D optimization

n = 2



- Your function should take as input:
  - The mathematical function to be minimized/optimized
  - A guess consisting of a point [x,y]
  - A set of 2 'direction vectors', each of size 2x1
  - Either **number of steps** or a **tolerance**.
- It should: Powell's method from Resources shows how to minimize along a direction
  - Make use of `from scipy.optimize import minimize_scalar`
  - Return a 2x1 array representing the solution as a point in the configuration space
  - Also return an array containing the entire history of your optimization search, i.e., every guess.
- Try it:
  - On a simple function such as $f([x, y]) = (x-1)^2 + (y+1)^2 + xy$
  - On the "noisy" function above from the Resources page.
- For a challenge:
  - Superimpose a line or curve and minimize with a constraint!