Kritik #9 Final

March 22, 2024

[4]:
```
#4a
import sympy as sp

x,y = sp.symbols ('x y') #defining variables
f= sp.exp(x) * sp.sin(y) + y**3 #defining functions
df_dx = sp.diff(f,x) #derivative in terms of x
df_dy = sp.diff(f,y) #derivative in terms of y

print("df/dx =", df_dx)
print("df/dy=", df_dy)
```

```
df/dx = exp(x)*sin(y)
df/dy= 3*y**2 + exp(x)*cos(y)
```

[7]:
```
#4b
import sympy as sp

w,z= sp.symbols ('w z') #defining variables
f= (w**2)*z + w*z**2 #defining function
df_dw = sp.diff(f,w) #partial derivative in terms of r
df_dz = sp.diff(f,z) #partial derivative in terms of t
coord = [(w,1), (z,-1)] #setting the point that we want to take derivative of
gradient_vector = [df_dw.subs(coord),df_dz.subs(coord)] #coordinates of gradient
#finding magnitude of gradient:
magnitude = sp.sqrt((df_dw.subs(coord)**2) + (df_dz.subs(coord)**2))


print ("Coordinates of gradient vector is",gradient_vector)
print ("Magnitude of gradient vector is",magnitude)
```

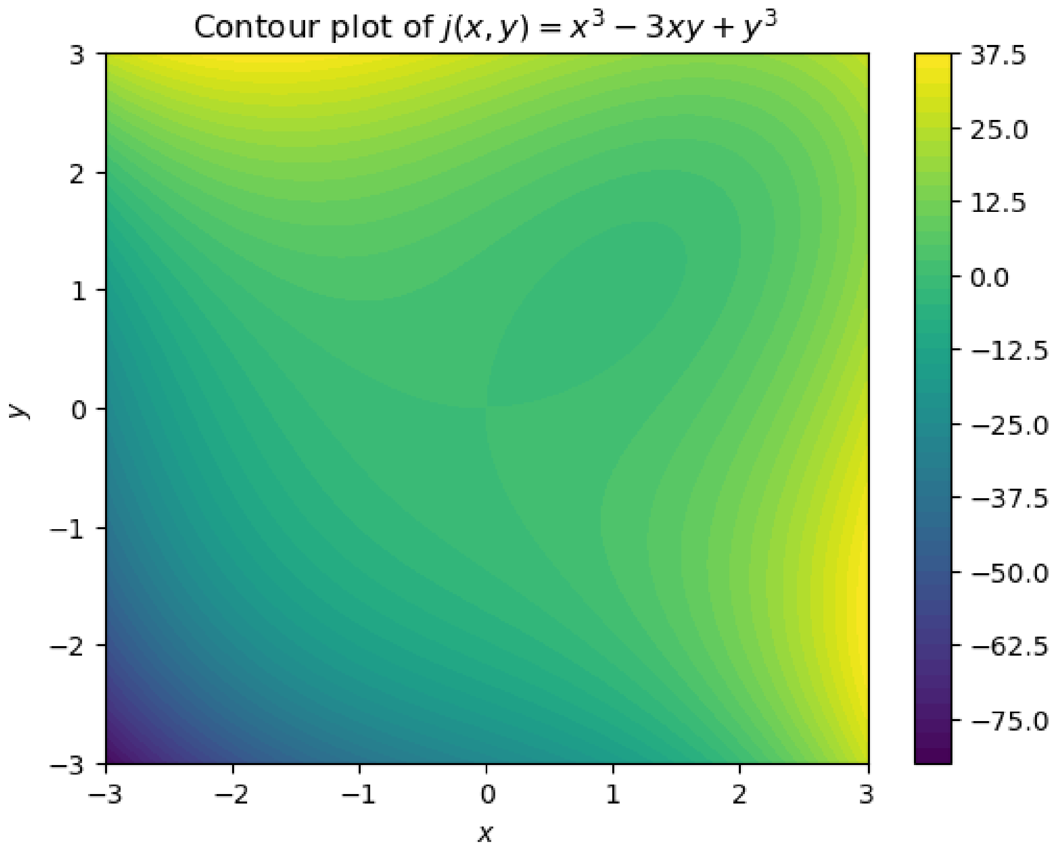
```
Coordinates of gradient vector is [-1, -1]
Magnitude of gradient vector is sqrt(2)
```

[8]:
```
#4c

#defining variables and function
x,y = sp.symbols ('x y')
f= sp.log(x**2 + y**2)
```

```
#finding second derivatives:
second_deriv_x = sp.diff(sp.diff(f,x),x)
second_deriv_y = sp.diff (sp.diff(f,y),y)
mixed_deriv = sp.diff (f,x,y)

print("Second deriv in terms of x =", second_deriv_x)
print("Second deriv in terms of y=", second_deriv_y)
print ("Mixed derivative =", mixed_deriv)
```

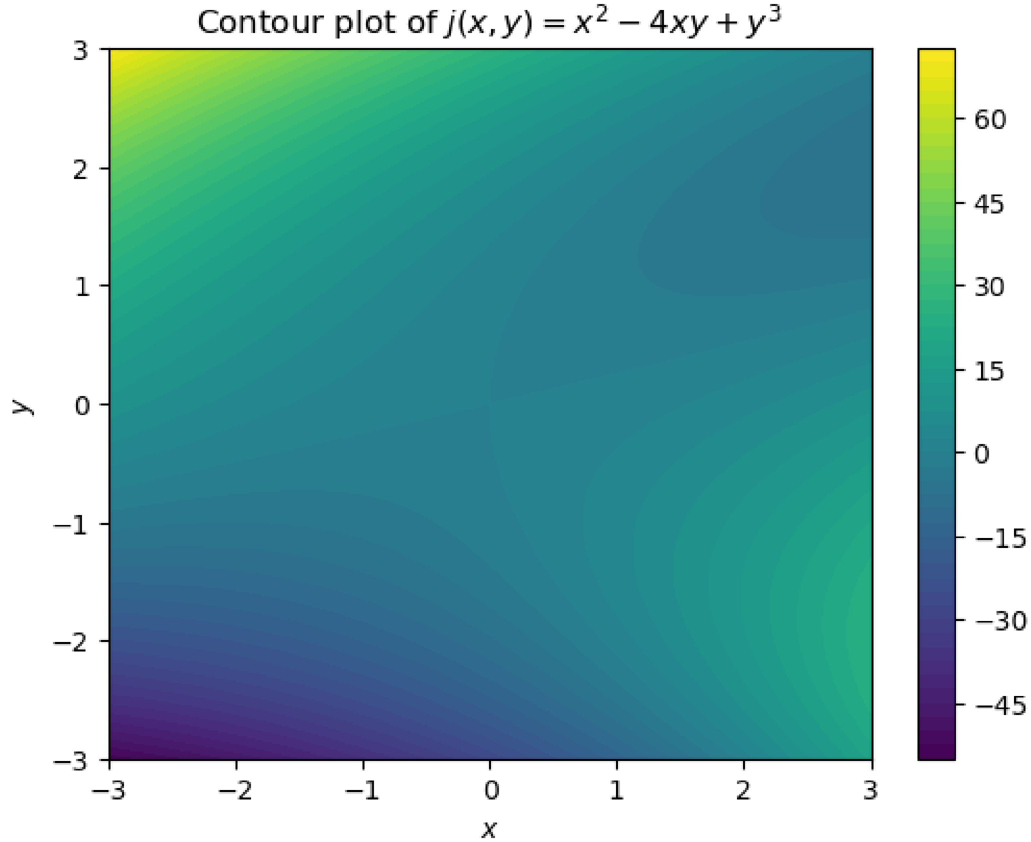
```
Second deriv in terms of x = -4*x**2/(x**2 + y**2)**2 + 2/(x**2 + y**2)
Second deriv in terms of y= -4*y**2/(x**2 + y**2)**2 + 2/(x**2 + y**2)
Mixed derivative = -4*x*y/(x**2 + y**2)**2
```

[21]:
```
#4d Greg's code
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
x, y = sp.symbols('x y')
j = x**3 - 3*x*y + y**3
j_func = sp.lambdify((x, y), j, 'numpy')
x_vals = np.linspace(-3, 3, 400)
y_vals = np.linspace(-3, 3, 400)
X, Y = np.meshgrid(x_vals, y_vals)
Z = j_func(X, Y)
plt.contourf(X, Y, Z, levels=50, cmap='viridis')
plt.colorbar()
plt.title('Contour plot of $j(x, y) = x^3 - 3xy + y^3$')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.show()
```



[23]:

```
#4d My graph
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
x, y = sp.symbols('x y')
j = x**2 - 4*x*y + y**3 #changed the function from c
j_func = sp.lambdify((x, y), j, 'numpy')
x_vals = np.linspace(-3, 3, 400)
y_vals = np.linspace(-3, 3, 400)
X, Y = np.meshgrid(x_vals, y_vals)
Z = j_func(X, Y)
plt.contourf(X, Y, Z, levels=50, cmap='viridis')
plt.colorbar()
plt.title('Contour plot of $j(x, y) = x^2 - 4xy + y^3$')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.show()
```


Contour plot of $j(x, y) = x^2 - 4xy + y^3$
3
2
1
0
−1
−2
−3
y
−3
−2
−1
0
1
2
3
x
60
45
30
15
0
−15
−30
−45

```
[ ]:
```