Assignment 10

February 9, 2022

1 Q1.

Investigate the nature of critical points for the following functions

$$1. f(x,y) = x^3 - 3x^2 + y^2$$

Q.1.1)

$f(x,y) = x^3 - 3x^2 + y^2$

$\nabla f = \begin{bmatrix} \partial f/\partial x \\ \partial f/\partial y \end{bmatrix} = \begin{bmatrix} 3x^2 - 6x \\ 2y \end{bmatrix}$

For critical points, we know $\nabla f = 0 \Rightarrow 3x^2 - 6x = 0$ & $2y = 0$

$\Rightarrow x = 2, 0$ & $y = 0$

$\Rightarrow (2,0)$ & $(0,0)$ are critical points.

Hessian $= \begin{bmatrix} \partial^2 f/\partial x^2 & \partial^2 f/\partial x \partial y \\ \partial^2 f/\partial y \partial x & \partial^2 f/\partial y^2 \end{bmatrix}$

$= \begin{bmatrix} \partial/\partial x (3x^2 - 6x) & \partial/\partial x (2y) \\ \partial/\partial y (3x^2 - 6x) & \partial/\partial y (2y) \end{bmatrix}$

Hessian $= \begin{bmatrix} 6x - 6 & 0 \\ 0 & 2 \end{bmatrix}$

At $(2,0)$, Hessian $= \begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$ eig vals $= m \pm \sqrt{m^2 - p}$

$= 4 \pm \sqrt{16 - 12}$

$= 4 \pm 2 \Rightarrow$ eigvals $= [6, 2]$

Actually, no need to calc. eig vals $\because$ We're still on x & y axes by this transformation and hence naturally, they should've been the leading diagonal elements, lol!

$\because$ At $(2,0)$, Hessian eigvals $> 0 \Rightarrow (2,0)$ is the minima

At $(0,0)$, Hessian $= \begin{bmatrix} -6 & 0 \\ 0 & 2 \end{bmatrix}$ eig vals $= [-6, 2]$

$\because$ Eig values of Hessian are positive & negative $\Rightarrow (0,0)$ is saddle point.

Ans] For $f = x^3 - 3x^2 + y^2$, $(2,0) \rightarrow$ Critical point of minimum $= -4$ (func. val)

$(0,0) \rightarrow$ Critical saddle point $= 0$ (func. val)

$$2. f(x,y) = x^2 + xy + y^2 + \frac{1}{x} + \frac{1}{y}, x \neq 0, y \neq 0$$

Q.1.2.

$f: x^2+y^2+xy+\frac{1}{x}+\frac{1}{y} \qquad x,y \neq 0$

$\nabla f = 0 \Rightarrow \begin{bmatrix} 2x+y-1/x^2 \\ 2y+x-1/y^2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Solve simultaneously

$\Rightarrow 2x+y-\frac{1}{x^2} = 0$ — ① & $2y+x-\frac{1}{y^2} = 0$ — ②

$\Rightarrow x = \frac{1}{y^2} - 2y = \frac{1-2y^3}{y^2}$

Substitute value of x from 2 in ①,

$2\left[\frac{1-2y^3}{y^2}\right] + y - \left(\frac{y^2}{1-2y^3}\right)^2 = 0$

$2\cancel{y}(1-2y^3)^3 + y^3(1-2y^3)^2 - y^6\cancel{(1-2y^3)^2} = 0$

$\therefore \quad 2\cancel{y}(1-6y^3+12y^6-8y^9)$

$+ \quad y^3(1-4y^3+4y^6)$

$-y^6 = 0$

$\Rightarrow 2 - 12y^3 + 24y^6 - 16y^9$

$+ y^3 - 4y^6 + 4y^9$

$-y^6 \qquad = 0$

$\Rightarrow 2 - 11y^3 + 19y^6 - 12y^9 = 0$

$\Rightarrow 12y^9 - 19y^6 + 11y^3 - 2 = 0$

Let $y^3 = t \Rightarrow 12t^3 - 19t^2 + 11t - 2 = 0$

$\Rightarrow t = \frac{1}{3}$ or $0.625 \pm 0.331i$

f is real-valued $\Rightarrow t = 1/3 \Rightarrow y^3 = 1/3 \Rightarrow y = 1/3^{1/3}$

Subst. y in ②,

$x = \frac{1-2y^3}{y^2} = y\cdot\left[\frac{1-2y^3}{y^3}\right] = y\cdot\left[\frac{1-2/3}{1/3}\right] = y$

$\therefore x = y \Rightarrow \left(\frac{1}{3^{1/3}}, \frac{1}{3^{1/3}}\right)$ is the critical point

$$\text{Hessian} = \begin{bmatrix} \partial^2 f/\partial x^2 & \partial^2 f/\partial x\,\partial y \\ \partial^2 f/\partial y\,\partial x & \partial^2 f/\partial y^2 \end{bmatrix}$$

$$= \begin{bmatrix} 2 + 2/x^3 & 1 \\ 1 & 2 + 2/y^3 \end{bmatrix}$$

$$\text{Hessian}\Big|_{(1/\sqrt[3]{3},\, 1/\sqrt[3]{3})} = \begin{bmatrix} 2 + 2\cdot(3) & 1 \\ 1 & 2 + 2(3) \end{bmatrix}$$

$$= \begin{bmatrix} 8 & 1 \\ 1 & 8 \end{bmatrix}$$

$$\text{Eigen values} = 8 \pm \sqrt{64 - 63}$$

$$= 8 \pm 1 = (9, 7)$$

Both eigen values $> 0 \Rightarrow$ Critical point is a minimum.

$\therefore$ Function attains a low at the point $\left(\frac{1}{\sqrt[3]{3}}, \frac{1}{\sqrt[3]{3}}\right)$

The func value is $\left(\frac{1}{\sqrt[3]{3}}\right)^2 \cdot 3 + \frac{2}{1/\sqrt[3]{3}} = 3^{4/3}$

2 Q2

Using Lagrange Multipliers, show that

a. The maximum value of $x^2y^3z^4$ subject to the constraint $2x + 3y + 4z = a$ is $(a/9)^9$

Q.2.1)

$f = x^2y^3z^4$ $\qquad\qquad g = 2x+3y+4z-a$

$$\nabla f = \begin{bmatrix} 2xy^3z^4 \\ 3x^2y^2z^4 \\ 4x^2y^3z^3 \end{bmatrix} \qquad \nabla g = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\nabla f = \lambda_0 \nabla g \Rightarrow \begin{bmatrix} 2x_0y_0^3z_0^4 \\ 3x_0^2y_0^2z_0^4 \\ 4x_0^2y_0^3z_0^3 \end{bmatrix} = \begin{bmatrix} 2\lambda \\ 3\lambda \\ 4\lambda \end{bmatrix}$$

$$\Rightarrow \lambda = x_0y_0^3z_0^4 = x_0^2y_0^2z_0^4 = x_0^2y_0^3z_0^3$$

$$\Rightarrow \lambda = x_0 = y_0 = z_0 = 1$$

$\therefore$ Optimal value of $f = x^2y^3z^4 = 1$

Since, $x=y=z=1$ lies on both f & g it could be shown or it follows that $(1,1,1)$ satisfies g

$$\Rightarrow 2(1)+3(1)+4(1) = a \Rightarrow a = 9 \Rightarrow \frac{a}{9} = 1 \Rightarrow \left(\frac{a}{9}\right)^9 = 1$$

$\therefore$ Optimal value of f s.t. constraint g is $\left(\frac{a}{9}\right)^9 = 1$.

Q.2.1] Continued

$$\text{Hessian} = \left.\begin{bmatrix} \partial/\partial x\,(2xy^3z^4) & \partial/\partial x\,(3x^2y^2z^4) & \partial/\partial x\,(4x^2y^3z^3) \\ \partial/\partial y\,(2xy^3z^4) & \partial/\partial y\,(3x^2y^2z^4) & \partial/\partial y\,(4x^2y^3z^3) \\ \partial/\partial z\,(2xy^3z^4) & \partial/\partial z\,(3x^2y^2z^4) & \partial/\partial z\,(4x^2y^3z^3) \end{bmatrix}\right|_{(1,1,1)}$$

$$= \left.\begin{bmatrix} 2y^3z^4 & 6xy^2z^4 & 8xy^3z^3 \\ 6xy^2z^4 & 6x^2yz^4 & 12x^2y^2z^3 \\ 8xy^3z^3 & 12x^2y^2z^3 & 12x^2y^3z^2 \end{bmatrix}\right|_{(1,1,1)}$$

$$= \begin{bmatrix} 2 & 6 & 8 \\ 6 & 6 & 12 \\ 8 & 12 & 12 \end{bmatrix} \qquad \text{Eigvals} = 25.6,\ -3.39,\ -2.20$$

$\Rightarrow (1,1,1)$ is a saddle point.

b. The minimum value of $yz + zx + xy$ subject to the constraint $xyz = a^2(x + y + z)$ is $9a^2$

Q.2.2] $f = yz + zx + xy$ $\qquad$ $g: xyz = a^2(x+y+z)$

$$\Rightarrow g: 1 = a^2\left(\frac{1}{yz} + \frac{1}{xz} + \frac{1}{xy}\right)$$

$$\Rightarrow g: 1 - a^2\left(\frac{1}{yz} + \frac{1}{xz} + \frac{1}{xy}\right)$$

Let us transform the variables s.t. $xy = p$, $yz = q$; $xz = r$

Then our problem becomes

$f = p + q + r$ $\qquad$ $g: 1 - a^2\left(\frac{1}{p} + \frac{1}{q} + \frac{1}{r}\right)$

Now $\nabla f = \lambda \Delta g$

$$\Rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} 0 + a^2/p^2 \\ 0 + a^2/q^2 \\ 0 + a^2/r^2 \end{bmatrix} \Rightarrow p^2, q^2, r^2 = a^2/\lambda \Rightarrow p, q, r = a/\sqrt{\lambda}$$

↳ Critical point

Q.2.2]

Substituting critical point in function, we get

optima $= p + q + r = \frac{3a}{\sqrt{\lambda}}$

But we know, the f^n g also satisfied by this point

$$\Rightarrow 1 = a^2\left(\frac{1}{p} + \frac{1}{q} + \frac{1}{r}\right)$$

$$\Rightarrow 1 = a^2\left(\frac{1}{a/\sqrt{\lambda}} + \frac{1}{a/\sqrt{\lambda}} + \frac{1}{a/\sqrt{\lambda}}\right) \Rightarrow 1 = 3a\sqrt{\lambda}$$

$$\Rightarrow \frac{1}{\sqrt{\lambda}} = 3a$$

Substituting in optima sol^n,

optimal value $= \frac{3a}{\sqrt{\lambda}} = 3a\left(\frac{1}{\sqrt{\lambda}}\right) = 3a \cdot 3a = 9a^2$

$$\text{Hessian} = \begin{bmatrix} \partial/\partial p\,(1) & \partial/\partial p\,(1) & \partial/\partial p\,(1) \\ \partial/\partial q\,(1) & \partial/\partial q\,(1) & \partial/\partial q\,(1) \\ \partial/\partial r\,(1) & \partial/\partial r\,(1) & \partial/\partial r\,(1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Rightarrow \text{No eigen values } (\because \text{Eig vec} \neq 0)$$

3 Q3

Find the minimum of $f(x, y) = \alpha x^2 + \beta y^2$ for various values of α, β, by

a. computing the gradient of f, τ

Q.3] $f(x,y) = \alpha x^2 + \beta y^2$

$$\psi(\tau) = \begin{bmatrix} x \\ y \end{bmatrix} - \tau \begin{bmatrix} 2\alpha x \\ 2\beta y \end{bmatrix}$$

$$F(\tau) = f(\psi(\tau))$$

$$= \alpha\left[(1-2\tau\alpha)x\right]^2 + \beta\left[(1-2\tau\beta)y\right]^2$$

$$F'(\tau) = 0$$

$$\therefore \alpha\left[2(1-2\tau\alpha)x^2 \cdot (-2\alpha)\right] + \beta\left[2(1-2\tau\beta)y^2 \cdot (-2\beta)\right] = 0$$

$$\therefore -4\alpha^2 x^2 (1-2\tau\alpha) - 4\beta^2 y^2 (1-2\tau\beta) = 0$$

$$\therefore \alpha^2 x^2 (1-2\tau\alpha) + \beta^2 y^2 (1-2\tau\beta) = 0$$

$$\therefore \alpha^2 x^2 + \beta^2 y^2 - 2\tau\left[\alpha^3 x^2 + \beta^3 y^2\right] = 0$$

$$\boxed{\therefore \tau = \frac{\alpha^2 x^2 + \beta^2 y^2}{2\left[\alpha^3 x^2 + \beta^3 y^2\right]}}$$

b. coding the iterations in Python with initial values $x_0 = 3, y_0 = 4$ and using the stopping criteria as $|f(j+1) - f(j)| < \epsilon = 1e^{-6}$

Estimate the order of convergence by plotting the error against number of iterations for a few cases.

[1]:
```python
from typing import Tuple
import pandas as pd
import matplotlib.pyplot as plt
pd.options.display.float_format = "{:,.6f}".format

# Define the staring points as constants
START_X = 3.0
START_Y = 4.0
```

[2]:
```python
def compute_function_value(alpha:float, beta:float, x:float, y:float) -> float:
    """[Given the value of alpha, beta and the point, maps x,y in domain
    of f to another real number as per the definition of f]

    Args:
        alpha (float): [The co-efficient of x^2]
        beta (float): [The co-efficient of y^2]
```

```
        x (float): [x value]
        y (float): [y value]

    Returns:
        [float]: [Function value]
    """
    return alpha * (x ** 2) + beta * (y ** 2)
```

[3]:
```
def compute_tau(alpha: float, beta:float, x:float, y:float) -> float:
    """[Compute tau value for the given function based on the following
    psi(tau) = xi - tau * grad(f)
    F(psi) = f(psi(tau))
    dF/dtau = 0 -> Solve for tau in terms of x and y
    ]

    Args:
        alpha (float): [The co-efficient of x^2]
        beta (float): [The co-efficient of y^2]
        x (float): [x value]
        y (float): [y value]

    Returns:
        [float]: [Tau value at a point (x,y) for the given function]
    """
    return 0.5 * ((alpha **2) * (x **2) + (beta **2) * (y ** 2)) / ((alpha **␣
 ↪3) * (x ** 2) + (beta ** 3) * (y ** 2))
```

[4]:
```
def update_variables(alpha: float, beta: float, x: float, y:float) -> Tuple:
    """[Perform update step in the opposite direction as gradient with the␣
 ↪update parameter tau]

    Args:
        alpha (float): [The co-efficient of x^2]
        beta (float): [The co-efficient of y^2]
        x (float): [x value]
        y (float): [y value]

    Returns:
        Tuple: [Updated values of x,y after performing the step]
    """
    tau = compute_tau(alpha, beta, x, y)
    x_updated = x - 2 * alpha * tau * x
    y_updated = y - 2 * beta * tau * y
    return (x_updated, y_updated)
```

[5]:
```
def optimize_function(alpha: float, beta: float, epsilon: float = 1e-6, plot:
 ↪bool = True):
```

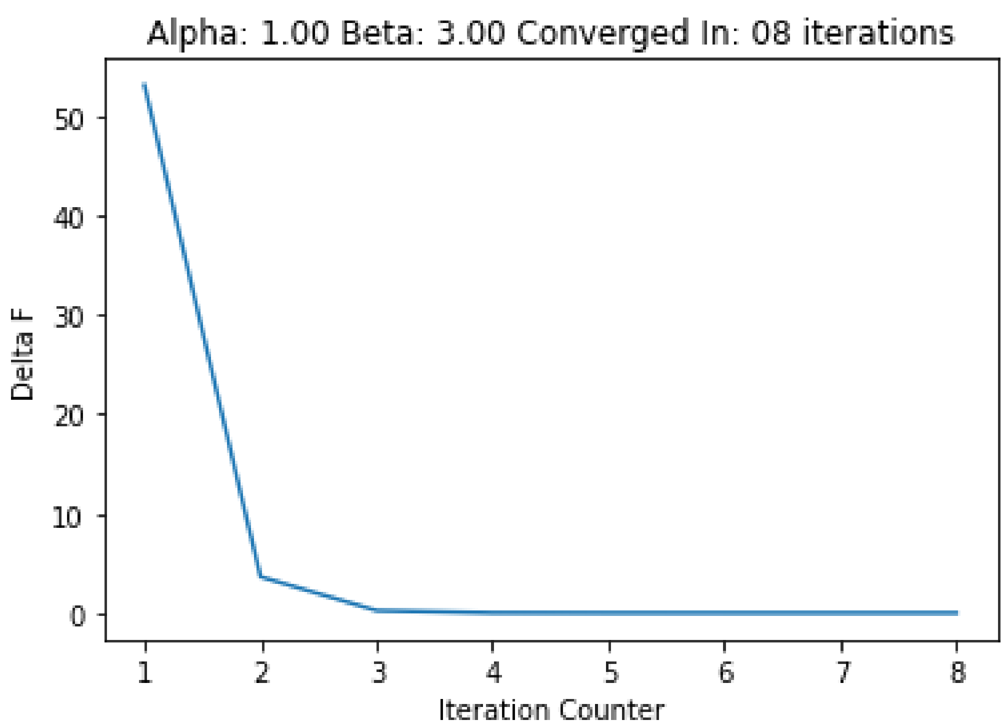
```python
    """[Optimize the function alpha*x^2 + beta*y^2 until the optimal solution␣
↪converges with a threshold of 1e-6]

    Args:
        alpha (float): [The co-efficient of x^2]
        beta (float): [The co-efficient of y^2]
        epsilon (float, optional): [Threshold of convergence]. Defaults to 1e-6.
        plot (bool, optional): [Whether to plot the progress of optimization␣
↪process]. Defaults to True.
    """

    # Define the starting parameters
    error_margin = 1
    max_iterations = 100
    iteration_counter = 0
    x, y = START_X, START_Y
    f_current = compute_function_value(alpha, beta, x, y)
    entries = []

    # Optimize the solutions
    while error_margin > epsilon:

        # Put a hard break on iterations exceeding a set threshold number
        if iteration_counter > max_iterations:
            break

        # Compute the current value of the function
        entries.append([x, y, f_current, error_margin])

        # Update the variables
        x, y = update_variables(alpha, beta, x, y)

        # Compute function value at the updated step and update the
        # stopping criterion
        f_new = compute_function_value(alpha, beta, x, y)
        error_margin = abs(f_new - f_current)

        # Make the new value to be the current value
        f_current = f_new

        # Increase the iteration counter
        iteration_counter += 1

    # Add the last entry
    entries.append([x, y, f_current, error_margin])
```

```
    # Create a summary dataframe and plot the function value and relative error
↪at each point
    # During the execution of the optimization
    execution_summary = pd.DataFrame(entries, columns = ["x", "y",
↪"function_value", "delta_f"])
    plt.plot(execution_summary.delta_f.iloc[1:], linestyle = "solid")
    plt.title(f"Alpha: {alpha:.2f} Beta: {beta:.2f} Converged In:
↪{iteration_counter:02d} iterations")
    plt.ylabel("Delta F")
    plt.xlabel("Iteration Counter")
    print(execution_summary)
```

Let $f = x^2 + 3y^2$

[6]:
```
optimize_function(1, 3)
```

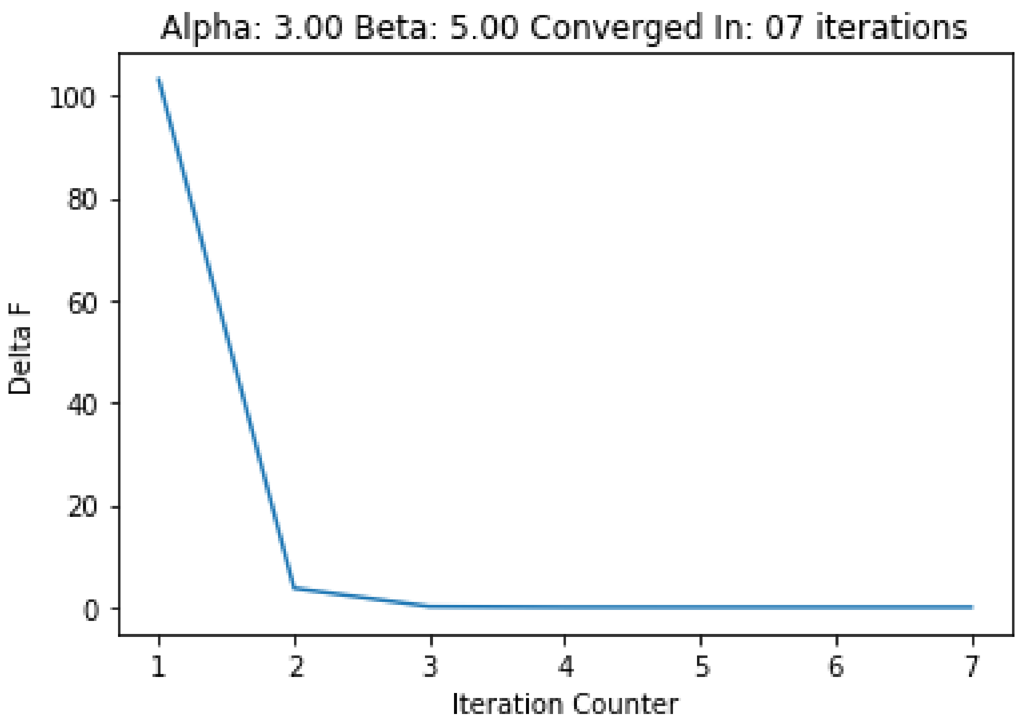
```
          x         y  function_value    delta_f
0  3.000000  4.000000       57.000000   1.000000
1  1.959184 -0.163265        3.918367  53.081633
2  0.206230  0.274973        0.269361   3.649006
3  0.134681 -0.011223        0.018517   0.250845
4  0.014177  0.018903        0.001273   0.017244
5  0.009258 -0.000772        0.000088   0.001185
6  0.000975  0.001299        0.000006   0.000081
7  0.000636 -0.000053        0.000000   0.000006
8  0.000067  0.000089        0.000000   0.000000
```



Let $f = 3x^2 + 5y^2$

[7]:
```
optimize_function(3, 5)
```

```
          x         y  function_value     delta_f
0 3.000000  4.000000      107.000000    1.000000
1 1.069996 -0.288899        3.851984  103.148016
2 0.108000  0.143999        0.138671    3.713313
3 0.038520 -0.010400        0.004992    0.133679
4 0.003888  0.005184        0.000180    0.004812
5 0.001387 -0.000374        0.000006    0.000173
6 0.000140  0.000187        0.000000    0.000006
7 0.000050 -0.000013        0.000000    0.000000
```

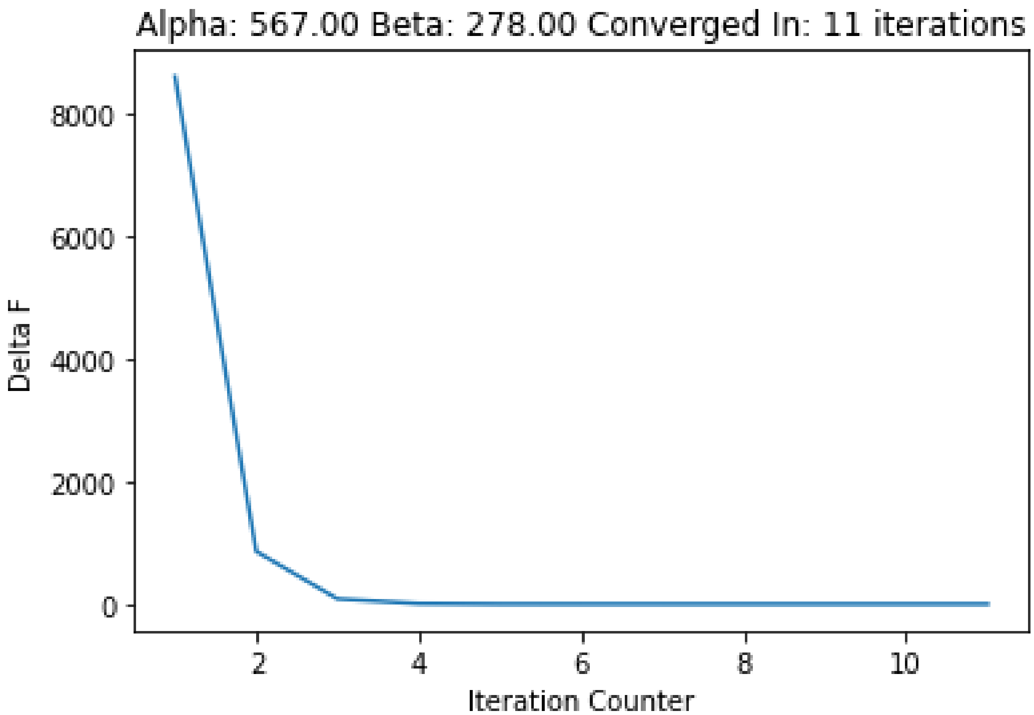


Let $f = 567^2 + 278y^2$

[8]:
```
optimize_function(567, 278)
```

```
           x         y  function_value       delta_f
0   3.000000  4.000000    9,551.000000      1.000000
1  -0.540278  1.685603      955.377146  8,595.622854
2   0.300087  0.400116       95.565437    859.811709
```

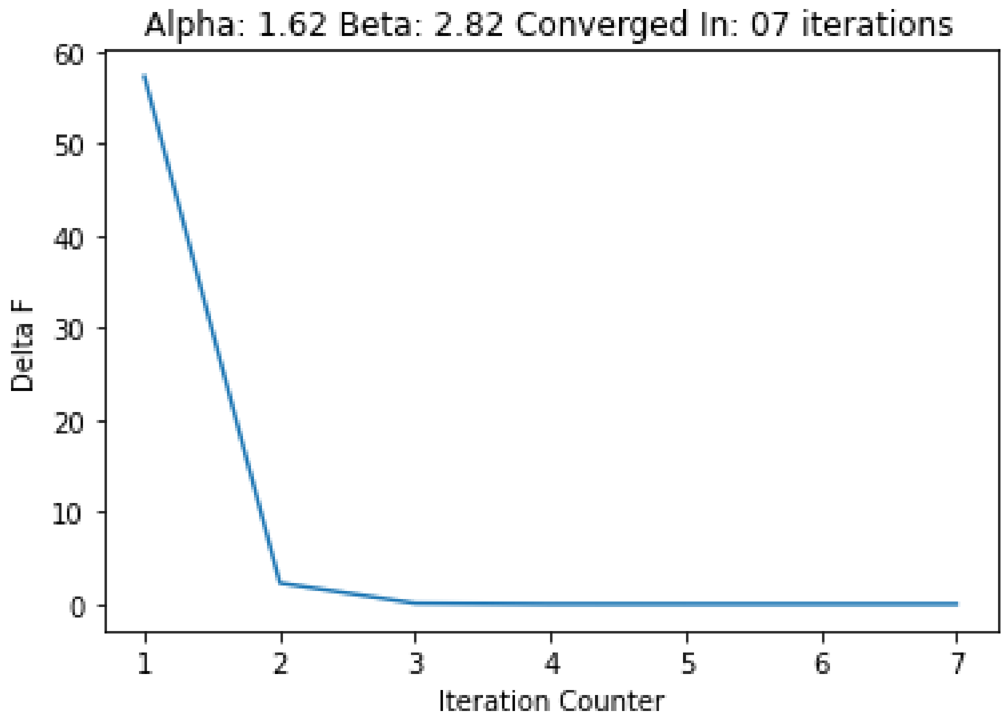
```
3   -0.054044 0.168609        9.559317    86.006120
4    0.030017 0.040023        0.956209     8.603108
5   -0.005406 0.016866        0.095649     0.860560
6    0.003003 0.004003        0.009568     0.086081
7   -0.000541 0.001687        0.000957     0.008611
8    0.000300 0.000400        0.000096     0.000861
9   -0.000054 0.000169        0.000010     0.000086
10   0.000030 0.000040        0.000001     0.000009
11  -0.000005 0.000017        0.000000     0.000001
```



Let $f = 1.618x^2 + 2.81828y^2$

[9]:
```
optimize_function(1.618, 2.81828)
```

```
          x         y  function_value   delta_f
0 3.000000  4.000000       59.654480  1.000000
1 1.154760 -0.285458        2.387206 57.267274
2 0.120052  0.160069        0.095529  2.291677
3 0.046210 -0.011423        0.003823  0.091707
4 0.004804  0.006406        0.000153  0.003670
5 0.001849 -0.000457        0.000006  0.000147
6 0.000192  0.000256        0.000000  0.000006
7 0.000074 -0.000018        0.000000  0.000000
```



As we can see from the plots above, the order of convergence is around 10 iterations for a tolerance value of $1e - 6$ for a function of the form $f = \alpha x^2 + \beta y^2$