MAE 598,

HW 2

Prob 1. $f(x_1, x_2) = 2x_1^2 - 4x_1x_2 + 1.5x_2^2 + x_2$

Gradient $\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \partial f / \partial x_2 \end{bmatrix} = \begin{bmatrix} 4x_1 - 4x_2 \\ -4x_1 + 3x_2 + 1 \end{bmatrix}$

A stationary point has $\nabla f = 0$

$\therefore$ so $4x_1 - 4x_2 = 0$ & $-4x_1 + 3x_2 + 1 = 0$

$\Rightarrow 4x_1 = 4x_2$

$\Rightarrow x_1 = x_2$, Substituting

$-4x_1 + 3x_1 + 1 = 0 \Rightarrow -x_1 + 1 = 0 \Rightarrow x_1 = 1$.

So $x_1 = x_2 = 1$, $(1, 1)$ is a stationary point.

Now lets determine Hessian,

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix}$$

Lets calculate eigen values of H

$(H - \lambda I) = 0 \Rightarrow \left(\begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right) = 0$

$= \left(\begin{bmatrix} 4-\lambda & -4 \\ -4 & 3-\lambda \end{bmatrix} \right) = 0$

$\Rightarrow (4-\lambda)(3-\lambda) - 16 = 0$

$\Rightarrow 12 + \lambda^2 - 7\lambda - 16 = 0$

$\Rightarrow \lambda^2 - 7\lambda - 4 = 0$

$\Rightarrow$ Solving the quadratic gives

$\lambda = 7.531128, -0.531128$

Thus as one λ is +ve & other is -ve.

& $|H| = 12 - 16 = -4 \ (< 0)$

H is indefinite and $(1,1)$ is a saddle point.

Using the Taylor series

$$f(x) = f(x_0) + \frac{\partial f}{\partial x}\Big|_{x_0}(x-x_0) + \frac{\partial^2 f}{\partial x^2}\Big|_{x_0}(x-x_0)^2$$

as at $(1,1) = x_0$ is a stationary point $\frac{\partial f}{\partial x}\Big|_{x_0} = 0$

so $f(x) = f(1,1) + \frac{1}{2}(x-x_0)^T H|_{x_0}(x-x_0)$

$$\Rightarrow f(x) = f(1,1) + \frac{1}{2}\begin{bmatrix} x_1-1 & x_2-1 \end{bmatrix}\begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix}\begin{bmatrix} x_1-1 \\ x_2-1 \end{bmatrix}$$

$$f(1,1) = 2(1)^2 - 4(1)(1) + 1.5(1)^2 + 1$$
$$= 0.5$$

and let $x_1 - 1 = \partial x_1$ & $x_2 - 1 = \partial x_2$

$$f(x) - f(1,1) = \frac{1}{2}\begin{bmatrix} \partial x_1 & \partial x_2 \end{bmatrix}\begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix}\begin{bmatrix} \partial x_1 \\ \partial x_2 \end{bmatrix}$$

$$= f(1,1) = \frac{1}{2}\begin{bmatrix} \partial x_1 & \partial x_2 \end{bmatrix}\begin{bmatrix} 4\partial x_1 - 4\partial x_2 \\ -4\partial x_1 + 3\partial x_2 \end{bmatrix}$$

$$= f(1,1) + \frac{1}{2}\left(4\partial x_1^2 - 4\partial x_1\partial x_2 - 4\partial x_1\partial x_2 + 3\partial x_2^2\right)$$

$$f(x) - 0.5 + \left(2\partial x_1^2 - 4\partial x_1\partial x_2 + 1.5\partial x_2^2\right)$$

solving to $2\partial x_1^2 - 4\partial x_1\partial x_2 + 1.5\partial x_2^2$

dividing by $\partial x_2^2 \Rightarrow 2\frac{\partial x_1^2}{\partial x_2^2} - 4\frac{\partial x_1}{\partial x_2} + 1.5$

let $\partial x_1/\partial x_2 = u$

$$\Rightarrow 2u^2 - 4u + 1.5$$

$\Rightarrow$ using quadratic roots 1.5 & 0.5

$\frac{\partial x_1}{\partial x_2} = 1.5 \Rightarrow \partial x_1 = 1.5\partial x_2$

& $\frac{\partial x_1}{\partial x_2} = 0.5 \Rightarrow \partial x_1 = 0.5\partial x_2$

Rewriting the eqn.

$$f(x) = 0.5 + \left[\left(\partial x_1 - \frac{3}{2}\partial x_2\right)\left(\partial x_1 - \frac{\partial x_2}{2}\right)\right]$$

$a=1,\ b=3/2,\ c=1,\ d=0.5$

So now, as (1,1) is a saddle point, the direction of descent slopes

$$f(x) - f(1,1) = \left(\partial x_1 - \frac{3}{2}\partial x_2\right)\left(\partial x_1 - \frac{\partial x_2}{2}\right) < 0$$

Prob 2 a) Plane $x_1+2x_2+3x_3=1$ Point $(-1,0,1)$

We need to find projection of the point on the plane. so that distance from plane to point is minimum.

Rewriting the plane eqn as $x_1+2x_2+3x_3-1=0$

let (x_1, x_2, x_3) be the point on the plane then, distance from (x_1, x_2, x_3) to $(-1, 0, 1)$ is least,

so distance $= \sqrt{(x_1-(-1))^2 + (x_2-0)^2 + (x_3-1)^2}$

$$= \sqrt{(x_1+1)^2 + x_2^2 + (x_3-1)^2}$$

Where to minimize the distance so

our main eqn to minimize is $(x_1+1)^2+(x_2)^2+(x_3-1)^2$

Now, from plane eqn, we can write

$$x_1 = -2x_2 - 3x_3 + 1$$

Substituting this in main eqn,

$$\min (-2x_2 - 3x_3 + 1 + 1)^2 + (x_2)^2 + (x_3 - 1)^2 = f(x)$$

Now to find the stationary point for minimum $f(x)$,

$$G_1 = \frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2(-2x_2 - 3x_3 + 2)(-2) + 2x_2 \\ 2(-2x_2 - 3x_3 + 2)(-3) + 2(x_3 - 1) \end{bmatrix}$$

Putting $\frac{\partial f}{\partial x} = 0 \Rightarrow \begin{bmatrix} \partial f/\partial x_2 \\ \partial f/\partial x_3 \end{bmatrix} = 0$

$$\Rightarrow -4(-2x_2 - 3x_3 + 2) + 2x_2 = 0$$
$$= +8x_2 + 12x_3 - 8 + 2x_2 = 0$$
$$= 10x_2 + 12x_3 - 8 = 0 \Rightarrow 5x_2 + 6x_3 - 4 = 0 \quad (1)$$

$$\& \quad -6(-2x_2 - 3x_3 + 2) + 2x_3 - 2$$
$$= +12x_2 + 18x_3 - 12 + 2x_3 - 2$$
$$= 12x_2 + 20x_3 - 14 = 0 \Rightarrow 6x_2 + 10x_3 - 7 = 0 \quad (2)$$

(1) & (2) we have 2 eq 2 unknown, ~~& solving for~~
Solving for x_1 & x_2 in (1) & (2), we get.

$$x_2 = -1/7 = -0.14286$$
$$x_3 = 11/14 = 0.785714$$
$$\& \text{ as } x_1 = -2x_2 - 3x_3 + 1$$
$$x_1 = -1.07141$$

Thus, $(x_1, x_2, x_3) = (-1.07141, -0.14286, 0.78571)$

We can calculate the Hessian to check if we have an optimal solution.

$$H = \begin{bmatrix} \partial^2 f/\partial x_1^2 & \partial^2 f/\partial x_1 \partial x_2 \\ \partial^2 f/\partial x_2 \partial x_1 & \partial^2 f/\partial x_2^2 \end{bmatrix}$$

$$= \begin{bmatrix} 10 & 12 \\ 12 & 20 \end{bmatrix}$$

Now checking $(H - \lambda I) = 0$

$$\left(\begin{bmatrix} 10 & 12 \\ 12 & 20 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right) = 0$$

$$= \left| \begin{bmatrix} 10-\lambda & 12 \\ 12 & 20-\lambda \end{bmatrix} \right| = 0$$

$$(10-\lambda)(20-\lambda) - 12^2 = 0$$

$$\Rightarrow \lambda^2 - 30\lambda + 56 = 0$$

$$\lambda = 2, 28$$

as both $\lambda > 0$, His p.d. ~~not convex~~,

and so $f(x)$ is a convex solution with valid solution of $(x_1, x_2, x_3) = (-1.07141, -0.14286, 0.78571)$

Prob 3 Proof that hyperplan is a convex set

For a hyperplan $a^T x = c$, where a is the normal dist^n to the hyperplane and, c is constant and x is any $\in R^n$

So for hyperplane to be convex set,
if 2 vector x_1 & x_2 belong to hyperplan,

$\lambda x_1 + (1-\lambda) x_2$ should also be in hyperplane
for $\lambda \in [0,1]$

so $x_1 = [x_{11}, x_{12}, x_{13} \dots x_{1n}]$
& $x_2 = [x_{21}, x_{22}, x_{23} \dots x_{2n}]$

& so $a^T x_1 = c$

$\Rightarrow [a_1 x_{11} + a_2 x_{12} + a_3 x_{13} + \dots] = c$ ①

& $a^T x_2 = c$

$\Rightarrow [a_1 x_{21} + a_2 x_{22} + x_3 x_{23} + \dots] = c$ - ②

doing $a^T(\lambda x_1 + (1-\lambda) x_2)$

$= a^T(\lambda x_{11} + (1-\lambda) x_{21}, \lambda x_{12} + (1-\lambda) x_{22}, \dots)$

$= (a_1 \lambda x_{11} + a_1 (1-\lambda) x_{21} + a_2 \lambda x_{12} + a_2 (1-\lambda) x_{22} + \dots)$

$= \lambda (a_1 x_{11} + a_2 x_{12} + a_3 x_{13} + \dots) + (1-\lambda)(a_1 x_{21} + a_2 x_{22} + a_3 x_{23} + \dots)$

$=$ then from ① & ②

$= \lambda (c) + (1-\lambda)(c)$

$= \lambda c + c - \lambda c = c$.

thus, $a^T(\alpha x_1 + (1-\alpha)x_2) = c$, so

$\alpha x_1 + (1-\alpha)x_2$ also belongs in the hyperplane, so it is a convex set.

Prob 4 a) $h = (I, I_t) = \begin{cases} I_t/I & \text{if } I \le I_t \\ I/I_t & \text{if } I_t \le I \end{cases}$

as I_t is target intensity, I_t is constant and I is a variable.

so for $I < I_t$, $h = \frac{I_t}{I}$, this is in the form $y = \frac{c}{x}$

which is a hyperbola till $I = I_t$

& $H(h) = \frac{\partial}{\partial I}\left(\frac{\partial (I_t/I)}{\partial I}\right) = \frac{2I_t}{I^3}$

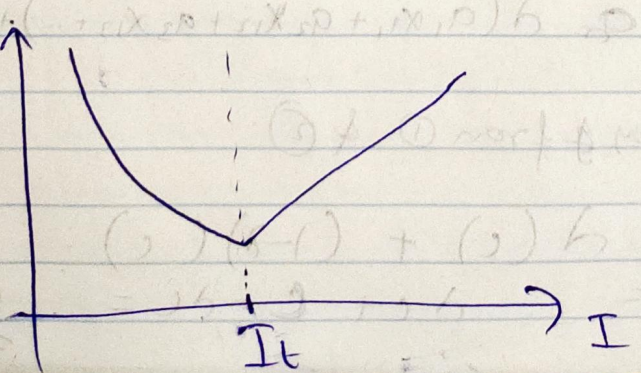
so as $P_i > 0$ for Power, $I \ge 0$ now,

so as $I \ge 0$, $\frac{2I_t}{I^3} > 0$, so it is convex for $I \le I_t$,

for $I > I_t$, $h = I/I_t$ which is of form $y = mx$

so, it is linear, and $H(h) \ge 0$, so it is also convex,

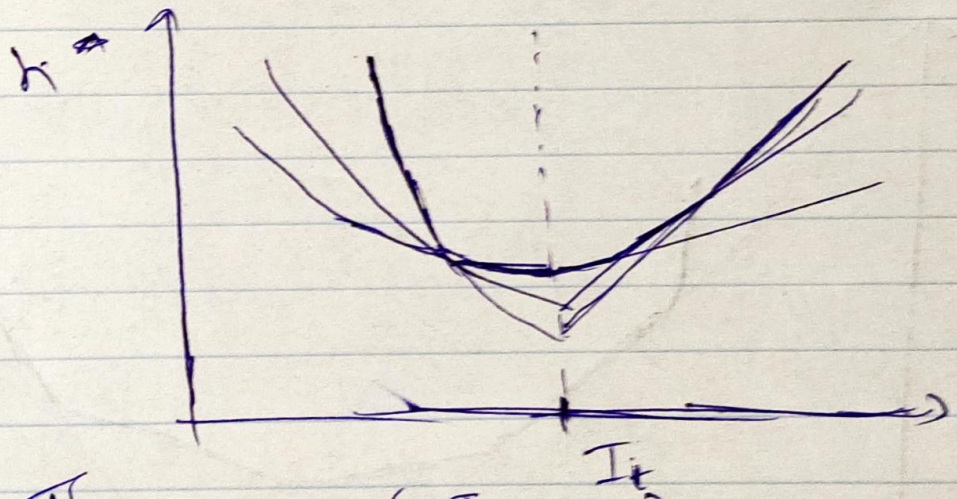
thus $h(I, I_t)$ is convex throughout.



Now I_k is defined as, $a_k^T p$, where a_k are the parameters for m mirrors. and the illumination problem is

$$\min_{p} \max_{k} \{ h(a_k^T p, I_k) \}$$

Thus we have to find the minimum value of maximum intersection of all the pos ip values of $h(\,)$.



Thus as all $h(a_k^T p, I_k)$ are convex function, max $(h())$ is a convex function as well

Prob 4 b) Overall power output of 10 lamps is less than P^*

$$\underbrace{P_1 + P_2 \ldots P_{10}}_{10 \text{ lamps.}} \leq P^*$$

But we can choose any of the 10 lamps out of n so we can choose it in C_n^{10} ways

Setting up linear constraints for this specific condn,

$$[1, 1, 1 \ldots 0 \ldots 0] \begin{bmatrix} P_1 \\ P_2 \\ \\ P_n \end{bmatrix} = P^*$$

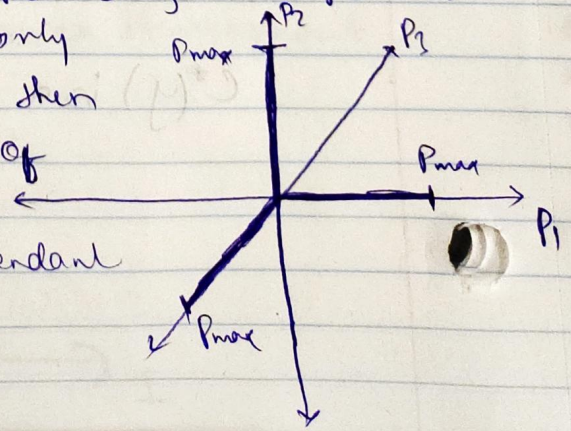
this is kind of a linear problem and can be solved with a unique solution and doesn't affect the solvability of the original minimization eqn.

c) Given that no more than 10 lamps are switched on, so the power output of any of the 10 lamps vary from 0 to P_{max}.

$$0 \leq \underbrace{P_1, P_2, P_3 \ldots P_{10}}_{10 \text{ lamps.}} \leq P_{max}$$

and the 10 lamps can be chosen in C_n^{10} ways.

But taking an example that only 3 lamps are available and only 1 lamp can be switched on, then we can see the possible set of solutions is shown in darkened bars along 3 independant axis



Suppose only 1 lamp can be turned on then we only choose one of the lamps as is chosen one axis but any other point in the space of $\mathbb{R}^3$ cannot be chosen as other lamps cannot be turned on, thus this proves that the set of input is not an convex set as an interpolation of the points in the set is not in the set. and thus we do not have a convex problem strictly and we may not have a unique solution.

Prob 5

The function to minimize is $xy - c(x)$ considering the function w.r.t. to y, the function is a linear in y as $c^*(y) = \max_x (xy - c(x))$

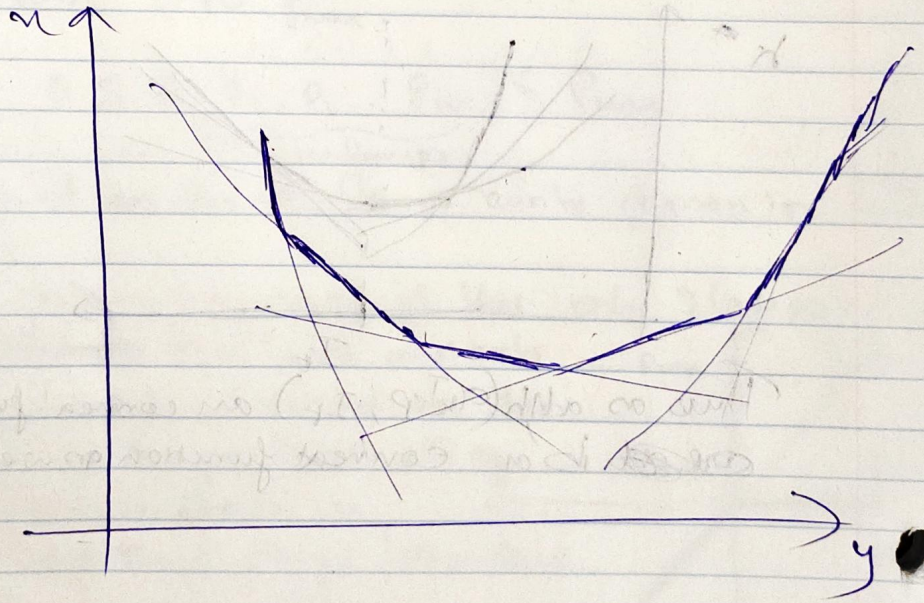
Any linear function is considered as a convex function as the Hessian of a linear function is 0, so H is psd. and can be considered as convex function.

Now as $xy - c(x)$ is convex w.r.t y

$\max (xy - c(x))$ can be evaluated as

$\max (x_1 y - c(x_1), x_2 y - c(x_2) \ldots)$,

as each of the functions are convex, the maximum of all of the functions is also a convex function. Thus, $c^*(y) = \max_x (xy - c(x))$ is a convex function.



MAE 598: Design Optimization ASU ID: 1223508715 HW2 Solution

Problem 2 b) Importing libraries to use

In [40]:
```
import numpy as np
from matplotlib import pyplot as plt
```

Defining functions for main function and Gradient function

In [41]:
```
def f(x):
  y=(-2*x[0]-3*x[1]+2)**2 + x[0]**2 + (x[1]-1)**2
  return y
```

In [42]:
```
def g(x):
  p=[
      10*x[0]+12*x[1]-8,
     12*x[0]+20*x[1]-14
  ]
  return np.array(p)
```

Initialising initial variables

In [43]:
```
t=0.5
alpha=1
x0 = np.array([6,7])
```

Algorithm for Gradient Descent using defined parameters

In [44]:
```
F=[]
F.append(f(x0))
G = np.linalg.norm(g(x0))

while G>1e-6:

  i=0
  while f(x0 - alpha*g(x0))>(f(x0)-t*alpha*np.matmul(np.transpose(g(x0)),g(x0))):
    alpha=0.5*alpha
    G=np.linalg.norm(g(x0))
    i += 1
    if i==100:
      break

  x0 = x0 - alpha*g(x0)
  F.append(f(x0))
  G = np.linalg.norm(g(x0))
  alpha = 1

F=np.array(F)

print('the final converged point is', x0)
```

```
the final converged point is [-0.14285689  0.78571413]
```

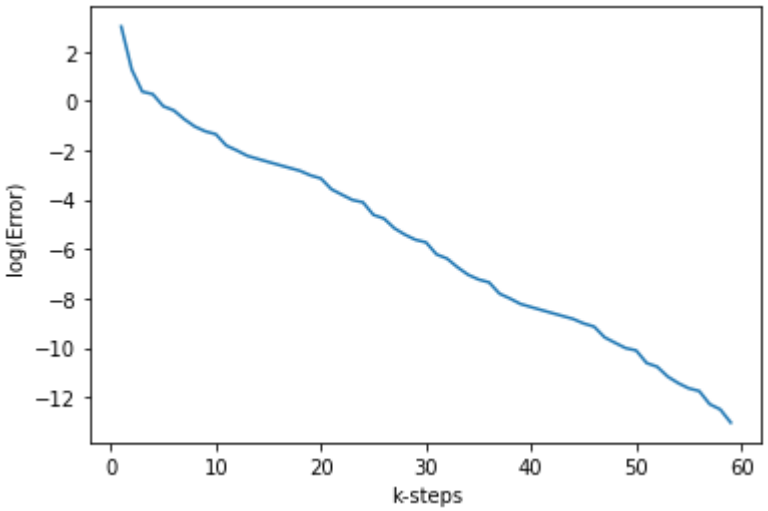
Defining the function for log linear plot using the calculated function values above and plotting

them.

In [45]:
```
def plotfunc(F):
  plt.plot(np.array([*range(1,len(F),1)]),np.log10(F[0:len(F)-1]-F[len(F)-1]))
  plt.xlabel('k-steps')
  plt.ylabel('log(Error)')
  return plt.show
```

In [46]:
```
plotfunc(F)
```

Out[46]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



Changing the initial point and again calculating the GD algorithm

In [47]:
```
t=0.5
alpha=1
x0 = np.array([1,1])

F=[]
F.append(f(x0))
G = np.linalg.norm(g(x0))

while G>1e-6:

  i=0
  while f(x0 - alpha*g(x0))>(f(x0)-t*alpha*np.matmul(np.transpose(g(x0)),g(x0))):
    alpha=0.5*alpha
    G=np.linalg.norm(g(x0))
    i += 1
    if i==100:
      break

  x0 = x0 - alpha*g(x0)
  F.append(f(x0))
  G = np.linalg.norm(g(x0))
  alpha = 1

F=np.array(F)

print('the final converged point for new parameters is', x0)
```

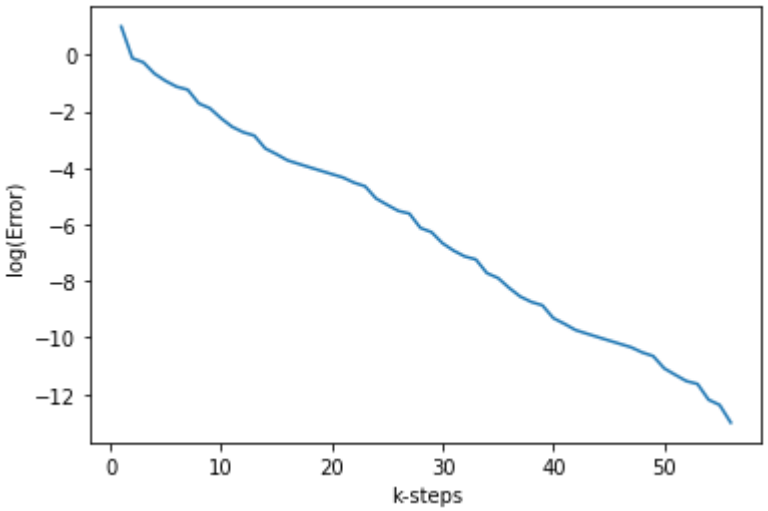
```
the final converged point for new parameters is [-0.14285686  0.78571409]
```

We can see that the final converged point is the same even after changing the initial points.

Plotting the log-linear graph for the new initial point

In [48]:
```
plotfunc(F)
```

Out[48]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



Again Defining new initial parameters and Hessian for the main function for implementing the Newton's method Note: the Hessian matrix is a constant matrix

In [49]:
```
t=0.5
alpha=1
x0 = np.array([13,21])
H=np.array([[10,12],[12,20]])
```

Algorithm for the Newton's method solver

In [50]:
```
F_new=[]
F_new.append(f(x0))
G = np.linalg.norm(g(x0))

while G>1e-6:

  i=0
  while f(x0 - alpha*np.matmul(np.linalg.inv(H),g(x0)))>(f(x0)-0.5*t*alpha*np.matmul(np
    alpha=0.5*alpha
    G=np.linalg.norm(g(x0))
    i += 1
    if i==100:
      break

  x0 = x0 - alpha*np.matmul(np.linalg.inv(H),g(x0))
  F_new.append(f(x0))
  G = np.linalg.norm(g(x0))
  alpha = 1

F_new=np.array(F_new)
```

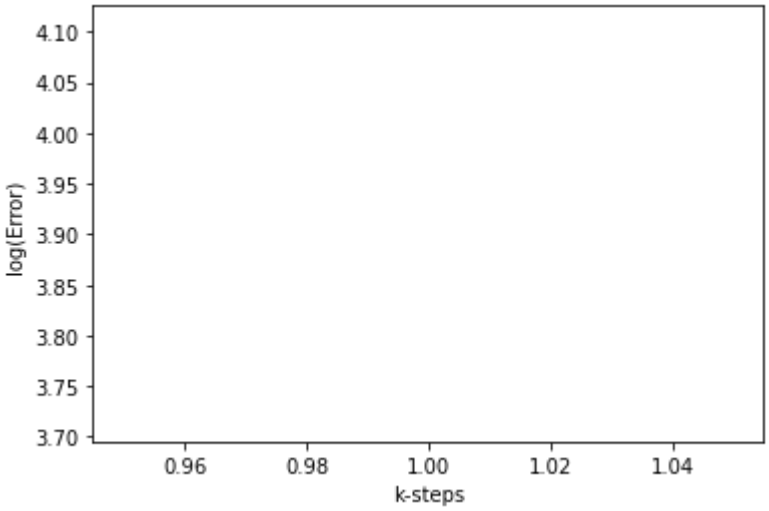
```
print('the final converged point is', x0)
```

```
the final converged point is [-0.14285714  0.78571429]
```

Plotting the log linear graph for convergence

In [51]:
```
plotfunc(F_new)
```

Out[51]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



Changing the inital point and calculating again

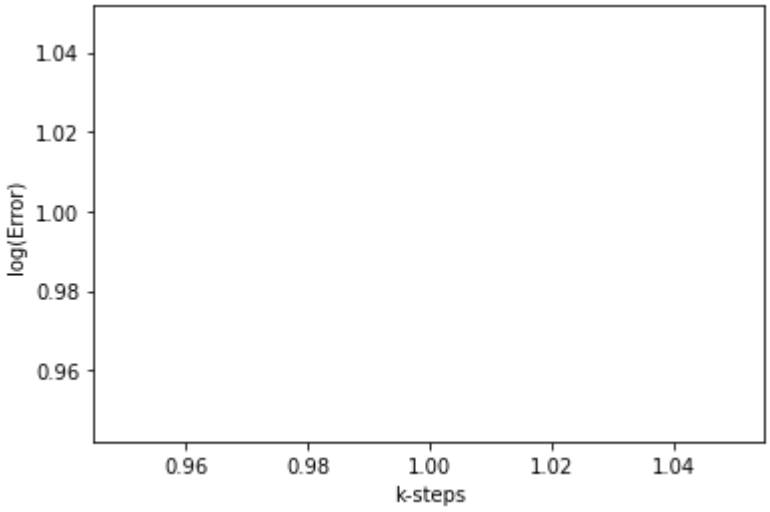
In [52]:
```
t=0.5
alpha=1
x0 = np.array([1,1])
H=np.array([[10,12],[12,20]])

F_new=[]
F_new.append(f(x0))
G = np.linalg.norm(g(x0))

while G>1e-6:

  i=0
  while f(x0 - alpha*np.matmul(np.linalg.inv(H),g(x0)))>(f(x0)-0.5*t*alpha*np.matmul(np
    alpha=0.5*alpha
    G=np.linalg.norm(g(x0))
    i += 1
    if i==100:
      break

  x0 = x0 - alpha*np.matmul(np.linalg.inv(H),g(x0))
  F_new.append(f(x0))
  G = np.linalg.norm(g(x0))
  alpha = 1

F_new=np.array(F_new)

print('the final converged point for new parameters is', x0)
```

```
the final converged point for new parameters is [-0.14285714  0.78571429]
```

Plotting the log linear graph for convergence

In [53]:
```
plotfunc(F_new)
```

Out[53]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



We can see that both the GD and Newton's method converge to the same point as calculated from the analytical way in 2 a) and both produce the expected log linear plot for error too.

In [53]:
```

```