Statistical Learning for Engineers (EN.530.641)
Homework 1

Jin Seob Kim, Ph.D.
Senior Lecturer, ME Dept., LCSR, JHU

Out: 09/09/2022
Due: 09/16/2022 by midnight EST

1 (a) In the class, we have discussed the properties that define a vector space. Let $\mathbf{x} \in \mathbb{R}^n$. Prove that

$$0\,\mathbf{x} = \mathbf{0}$$

only by using those properties. Do not use component form calculations. Here 0 denotes scalar zero, while $\mathbf{0}$ means the zero vector in $\mathbb{R}^n$.

(b) Using part (a), prove that $-\mathbf{x} = (-1)\mathbf{x}$. Note that through this, one can define the difference between two vectors $\mathbf{x}$, $\mathbf{y}$ in a vector space, i.e., $\mathbf{x} - \mathbf{y}$.

(c) Prove that $\mathbb{R}^{m\times n}$ (the set of all real $m \times n$ matrices) with matrix addition, $+$, and scalar multiplication, $\cdot$, forms a vector space.

2 (a) Let's consider a system of equations

$$\begin{pmatrix} 1 & 1 & 2 \\ 1 & -1 & 2 \\ 2 & 0 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}.$$

Does this system of equations have a solution? How do you know?

(b) Suppose that, for $\mathbb{R}^2$, we have a basis set $\{\mathbf{v}_1, \mathbf{v}_2\}$ that is not orthonormal. How do you obtain the orthonormal basis set? Can you generalize the process into $\mathbb{R}^n$?

3 In the class, we have learned FONC as $\mathbf{v}^T \nabla f(\mathbf{x}^*) \geq 0$ for a local minimizer $\mathbf{x}^*$ ($\mathbf{v}$ is a feasible direction). If $\mathbf{x}^*$ is located inside the domain, then FONC becomes

$$\nabla f(\mathbf{x}^*) = \mathbf{0}.$$

Prove that this is indeed true.

4 Given a function $f : \mathbb{R}^2 \to \mathbb{R}$ which is explicitly written as $f(x_1, x_2) = x_1^2 + 2x_1x_2 + x_2^3$,

(a) Compute the gradient ∇f.

(b) Compute the Hessian $D^2 f$.

(c) Expand $f(x_1, x_2)$ by using the Taylor series about the point $(1, 2)$ up to the second order.

$\frac{\partial f}{\partial x_1} = 200(x_2 - x_1^2) \times -2x_1 - 2(1-x_1)$

5 Let $f : \mathbb{R}^2 \to \mathbb{R}$ have the following form

$$f(\mathbf{x}) = f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

$-400(x_2 - x_1^2)x_1 - 2 + 2x_1$

where $\mathbf{x} = [x_1, x_2]^T$.

(a) Draw a contour plot using python. (Hint: look for `contourf` in numpy).

$-400x_2x_1 + 400x_1^3 - 2 + 2x_1$

(b) Can you find any minimum point? (do it analytically; you are to do it numerically in the next problem). Is it a local or a global minimizer?

$\frac{\partial f}{\partial x_2} = 200(x_2 - x_1^2)$

6 Now you are asked to write your python code for finding a minimum of the function given in Problem **5**. Choose any initial point that is not close to the minimum. Then implement the following algorithms:

(a) Gradient descent algorithm, where α_k is constant. You will have to play with this parameter.

(b) Newton's algorithm.

$$H = \begin{bmatrix} -400x_2 + 1200x_1^2 + 2 & -400x_1 \\ -400x_1 & 200 \end{bmatrix}$$

As the outcomes:

- Report min f and $\mathbf{x}^*$ (the corresponding minimum point). And compare it with the results in Problem **5** (b).
- Generate the plot of function values vs. step numbers.
- Generate plots that looks like Fig. 1.

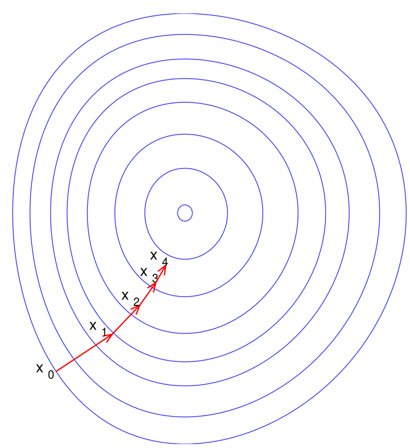


Figure 1: Problem **6**: An exemplary figure that shows the steps and movements during the gradient-based optimization process, with contour plot of the function. This is from wikipedia.

Submission Guideline

- Submit the analytic parts of the homework (Problem 1 to 5) in pdf format, including plots, to "HW1_analytical" on Gradescope.
  - No more than two (2) homework problems may be on the same page. In other words, for each problem your answers should be on a separate set of pages (e.g., for Problem 1, your answer is on page 1–3, and for Problem 2, your answer is on page 4-5, and so on).

- When submitting, you should assign the pages to each problem on Gradescope. You can scan your answers or use an app (e.g., Adobe Scan) to generate a pdf file. *Show your work.*

- Submit to "HW1_computational" on Gradescope a single zip file that contains all the codes for each computational problem (if any, codes for analytic parts as well). Name your single zip file as "YourName_HW1.zip". For example, "JinSeobKim_HW1.zip" for a single zip file. Submission will be done through the Gradescope.
- Just in case you have related separate files, please make sure to include *all the necessary files*. If TAs try to run your function and it does not run, then your submission will have a significant points deduction.
- Make as much comments as possible so that the TAs can easily read your codes.

1 (a) In the class, we have discussed the properties that define a vector space. Let $\mathbf{x} \in \mathbb{R}^n$. Prove that

$$0\,\mathbf{x} = \mathbf{0}$$

only by using those properties. Do not use component form calculations. Here 0 denotes scalar zero, while $\mathbf{0}$ means the zero vector in $\mathbb{R}^n$.

(b) Using part (a), prove that $-\mathbf{x} = (-1)\mathbf{x}$. Note that through this, one can define the difference between two vectors $\mathbf{x}$, $\mathbf{y}$ in a vector space, i.e., $\mathbf{x} - \mathbf{y}$.

(c) Prove that $\mathbb{R}^{m\times n}$ (the set of all real $m \times n$ matrices) with matrix addition, $+$, and scalar multiplication, $\cdot$, forms a vector space.

a) For any $X \in \mathbb{R}^n$. there $\exists$ $-X \in \mathbb{R}^n$. s.t $X + (-X) = 0$.

By distributively $0 = X + (-X) = (1 + (-1))X = 0 \cdot X$

b). For any $X \in \mathbb{R}^n$. $\exists$ $-X \in \mathbb{R}^n$ s.t $X + (-X) = 0$.

$X + (-X) = (1 + (-1))X = X + (-1)X$

$-X + X + (-X) = -X + X + (-1)X \Rightarrow 0X + (-X) = 0X + (-1)X \Rightarrow (-X) = (-1)X$

c). For any $A_{m\times n} \in \mathbb{R}^{m\times n}$. $B_{m\times n} \in \mathbb{R}^{m\times n}$. $\alpha \in \mathbb{R}$.

where $A = \begin{bmatrix} a_{11} & a_{12} \cdots a_{1n} \\ \vdots & \\ a_{m1} & \cdots a_{mn} \end{bmatrix}$ $B = \begin{bmatrix} b_{11} & b_{12} \cdots b_{1n} \\ \vdots & \\ b_{m1} & \cdots b_{mn} \end{bmatrix}$

① $A + B = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} \cdots a_{1n}+b_{1n} \\ \vdots & \\ a_{m1}+b_{m1} & \cdots a_{mn}+b_{mn} \end{bmatrix} \in \mathbb{R}^{m\times n}$.

② $\alpha A = \begin{bmatrix} \alpha a_{11} & \cdots \alpha a_{1n} \\ \vdots & \\ \alpha a_{m1} & \cdots \alpha a_{mn} \end{bmatrix} \in \mathbb{R}^{m\times n}$

$\Rightarrow \mathbb{R}^{m\times n}$ with matrix addition and multiplication forms a vector space.

Prove properties.

① $A, B \in \mathbb{R}^{m\times n}$ $\quad A+B=\begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ \vdots & & & \\ a_{m1}+b_{m1} & \cdots & & a_{mn}+b_{mn}\end{bmatrix} = \begin{bmatrix} b_{11}+a_{11} & b_{12}+a_{21} & \cdots & b_{1n}+a_{1n} \\ & & & \\ b_{m1}+a_{m1} & \cdots & & b_{mn}+a_{mn}\end{bmatrix} = B+A$ ✓

② $A, B, C \in \mathbb{R}^{m\times n}$.

$$(A+B)+C = \begin{bmatrix} (a_{11}+b_{11})+c_{11} & \cdots & (a_{1n}+b_{1n})+c_{1n} \\ \vdots & & \\ (a_{m1}+b_{m1})+c_{m1} & \cdots & (a_{mn}+b_{mn})+c_{mn}\end{bmatrix} = \begin{bmatrix} a_{11}+(b_{11}+c_{11}) & \cdots & a_{1n}+(b_{1n}+c_{1n}) \\ \vdots & & \\ a_{m1}+(b_{m1}+c_{m1}) & \cdots & a_{mn}+(b_{mn}+c_{mn})\end{bmatrix} = A+(B+C)$$

③ there $\exists\, 0$

s.t any $A \in \mathbb{R}^{m\times n}$

$$\begin{bmatrix} 0 & & \\ & & \\ & & \end{bmatrix} + \begin{bmatrix} a_{11} & a_{12} \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} = \begin{bmatrix} 0+a_{11} & \cdots & 0+a_{1n} \\ \vdots & & \\ 0+a_{m1} & \cdots & 0+a_{mn}\end{bmatrix} = A$$

④ ~~For~~ any $A \in \mathbb{R}^{m\times n}$. there exist $\begin{bmatrix} -a_{11} & \cdots & -a_{1n} \\ \vdots & & \\ -a_{m1} & \cdots & -a_{mn}\end{bmatrix}$ s.t

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} + \begin{bmatrix} -a_{11} & \cdots & -a_{1n} \\ \vdots & & \\ -a_{m1} & \cdots & -a_{mn}\end{bmatrix} = \begin{bmatrix} a_{11}-a_{11} & \cdots & a_{1n}-a_{1n} \\ \vdots & & \\ a_{m1}-a_{m1} & \cdots & a_{mn}-a_{mn}\end{bmatrix} = \begin{bmatrix} 0 & \cdots & 0 \\ & \vdots & \\ 0 & & 0\end{bmatrix} = 0$$

⑤ $$\alpha(A+B) = \alpha\left(\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} + \begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & & \\ b_{m1} & \cdots & b_{mn}\end{bmatrix}\right) = \alpha\begin{bmatrix} a_{11}+b_{11} & \cdots & a_{1n}+b_{1n} \\ \vdots & & \\ a_{m1}+b_{m1} & \cdots & a_{mn}+b_{mn}\end{bmatrix} = \begin{bmatrix} \alpha(a_{11}+b_{11}) & \cdots & \alpha(a_{1n}+b_{1n}) \\ \vdots & & \\ \alpha(a_{m1}+b_{m1}) & \cdots & \alpha(a_{mn}+b_{mn})\end{bmatrix} =$$

$$\begin{bmatrix} \alpha a_{11}+\alpha b_{11} & \cdots & \alpha a_{1n}+\alpha b_{1n} \\ & & \vdots \\ \alpha a_{m1}+\alpha b_{m1} & \cdots & \alpha a_{mn}+\alpha b_{mn}\end{bmatrix} = \begin{bmatrix} \alpha a_{11} & \cdots & \alpha a_{1n} \\ & & \\ \alpha a_{m1} & & \alpha a_{mn}\end{bmatrix} + \begin{bmatrix} \alpha b_{11} & \cdots & \alpha b_{1n} \\ \vdots & & \\ \alpha b_{m1} & \cdots & \alpha b_{mn}\end{bmatrix} = \alpha A + \alpha B$$

⑥ $$(\alpha+\beta)A = \begin{bmatrix} (\alpha+\beta)a_{11} & \cdots & (\alpha+\beta)a_{1n} \\ \vdots & & \vdots \\ (\alpha+\beta)a_{m1} & & (\alpha+\beta)a_{mn}\end{bmatrix} = \begin{bmatrix} \alpha a_{11}+\beta a_{11} & \cdots & \alpha a_{1n}+\beta a_{1n} \\ \vdots & & \\ \alpha a_{m1}+\beta a_{m1} & \cdots & \alpha a_{mn}+\beta a_{mn}\end{bmatrix} = \begin{bmatrix} \alpha a_{11} & \cdots & \alpha a_{1n} \\ & & \\ \alpha a_{m1} & & \alpha a_{mn}\end{bmatrix} + \begin{bmatrix} \beta a_{11} & \cdots & \beta a_{1n} \\ \vdots & & \\ \beta a_{m1} & \cdots & \beta a_{mn}\end{bmatrix} = \alpha A + \beta A.$$

⑦ $$(\alpha\beta)\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} = \begin{bmatrix} \alpha\beta a_{11} & \cdots & \alpha\beta a_{1n} \\ & & \\ \alpha\beta a_{m1} & \cdots & \alpha\beta a_{mn}\end{bmatrix} = \alpha\begin{bmatrix} \beta a_{11} & \beta a_{1n} \\ \beta a_{m1} & \beta a_{mn}\end{bmatrix} = \alpha(\beta A)$$

⑧ $$\exists I. \text{ s.t } \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1\end{bmatrix}\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn}\end{bmatrix} = A.$$

2 (a) Let's consider a system of equations

$$\begin{pmatrix} 1 & 1 & 2 \\ 1 & -1 & 2 \\ 2 & 0 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}.$$

Does this system of equations have a solution? How do you know?

(b) Suppose that, for $\mathbb{R}^2$, we have a basis set $\{\mathbf{v}_1, \mathbf{v}_2\}$ that is not orthonormal. How do you obtain the orthonormal basis set? Can you generalize the process into $\mathbb{R}^n$?

2 a) $$\left[\begin{array}{ccc|c} 1 & 1 & 2 & 3 \\ 1 & -1 & 2 & 2 \\ 2 & 0 & 4 & 1 \\ 2 & 2 & 4 & 6 \end{array}\right] \to \left[\begin{array}{ccc|c} 1 & 1 & 2 & 3 \\ 0 & 2 & 0 & 1 \\ 0 & 2 & 0 & 5 \end{array}\right] \to \left[\begin{array}{ccc|c} 1 & 1 & 2 & 3 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 2 \end{array}\right]$$

$\Rightarrow$ No solution. since $[0\ 0\ 0 \mid 2]$ is inconsistent.

b) $\vec{u_1} = \vec{v_1}$

$$\vec{u_2} = \vec{v_2} - \frac{\langle \vec{u_2} . \vec{v_1} \rangle}{|\vec{u_1}|^2} \vec{u_1}$$

Yes! $$\vec{u_k} = \vec{v_k} - \sum_{i=1}^{k-1} \frac{\langle \vec{u_k}, \vec{v_i} \rangle}{|\vec{u_i}|^2} \vec{u_i}$$

3 In the class, we have learned FONC as $\mathbf{v}^T \nabla f(\mathbf{x}^*) \geq 0$ for a local minimizer $\mathbf{x}^*$ ($\mathbf{v}$ is a feasible direction). If $\mathbf{x}^*$ is located inside the domain, then FONC becomes

$$\nabla f(\mathbf{x}^*) = \mathbf{0}.$$

Prove that this is indeed true.

If x^* is located inside the domain, then every direction is feasible,

$$\begin{cases} v^T \nabla f(x^*) \geq 0 \\ (-v)^T \nabla f(x^*) \geq 0 \end{cases} \Rightarrow \begin{matrix} v^T \nabla f(x^*) \geq 0 \\ v^T \nabla f(x^*) \leq 0 \end{matrix}$$

$$\Rightarrow \nabla f(x^*) = 0.$$

4 Given a function $f : \mathbb{R}^2 \to \mathbb{R}$ which is explicitly written as $f(x_1, x_2) = x_1^2 + 2x_1x_2 + x_2^3$,

(a) Compute the gradient ∇f.

(b) Compute the Hessian $D^2 f$.

(c) Expand $f(x_1, x_2)$ by using the Taylor series about the point $(1, 2)$ up to the second order.

a) $\nabla f = \begin{bmatrix} 2x_1 + 2x_2 \\ 2x_1 + 3x_2^2 \end{bmatrix}$

b) $D^2 f = \begin{bmatrix} \frac{\partial f}{\partial x_1^2} & \frac{\partial f}{\partial x_2 \partial x_1} \\ \frac{\partial f}{\partial x_1 \partial x_2} & \frac{\partial f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 2 & 2 \\ 2 & 6x_2 \end{bmatrix}$

c) $f(x+v) = f(x) + \langle \nabla f(x). v \rangle + \frac{1}{2} \langle v. D^2 f(x) v \rangle$

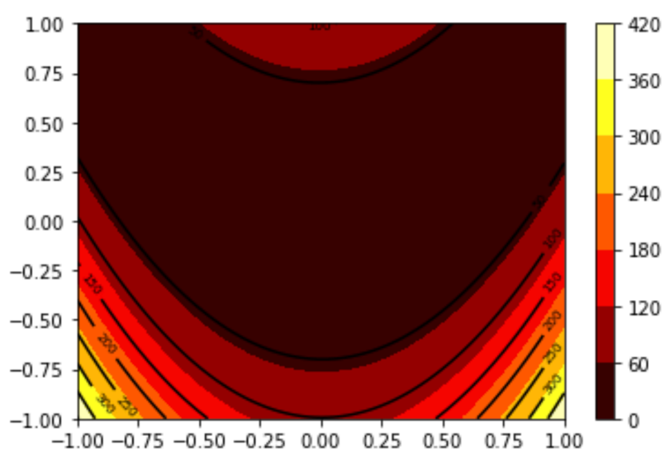
$f(x_1.x_2) = 1 + 2\times 2 + 2^3 = 1 + 4 + 8 = 13.$

$\nabla f(1.2) = \begin{bmatrix} 2 + 2\times 2 \\ 2 + 3\times 2^2 \end{bmatrix} = \begin{bmatrix} 6 \\ 14 \end{bmatrix}$

$D^2 f(x) = \begin{bmatrix} 2 & 2 \\ 2 & 12 \end{bmatrix}.$

$f(x+v) = 13 + 6v_1 + 14v_2 + 2v_1^2 + 4v_1v_2 + 12v_2^2$

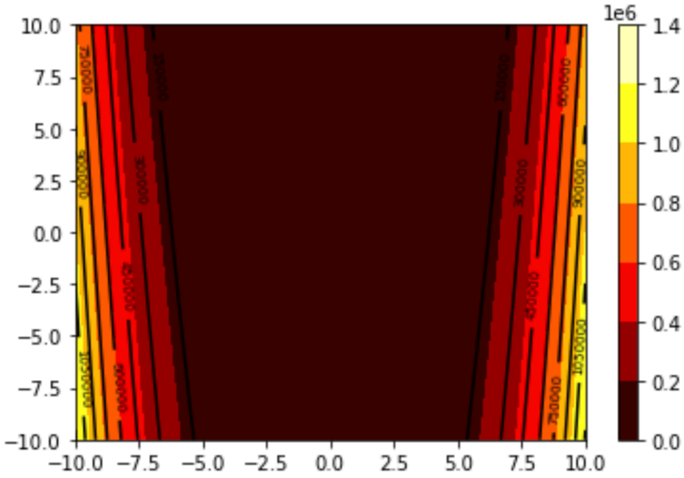
5.



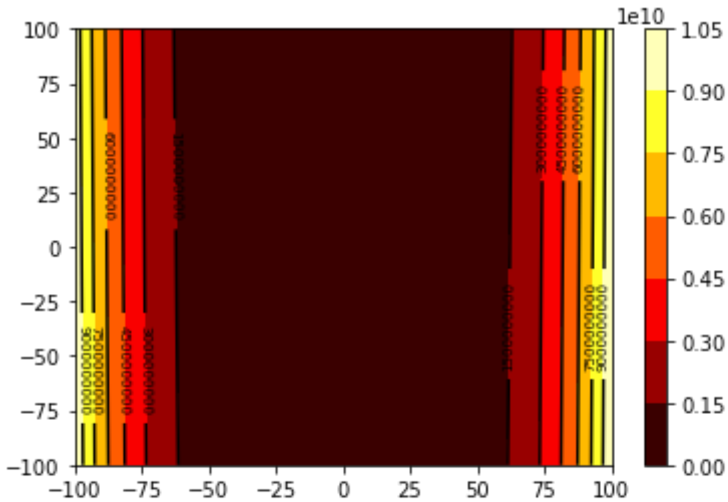
Yes.
$(0, 0.75)$
$(0, -0.6)$

In [19]:

```python
##5
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import cm

fig = plt.figure()
ax = fig.add_subplot(111)
u = np.linspace(-10, 10, 100)
x, y = np.meshgrid(u, u)
z = 100*(y-x**2)**2+(1-x)**2
cset = plt.contourf(x,y,z,6, cmap = plt.cm.hot)
contour = plt.contour(x,y,z,8, colors = 'k')
plt.clabel(contour, fontsize = 7, colors = 'k')
plt.colorbar(cset)
plt.show()
```

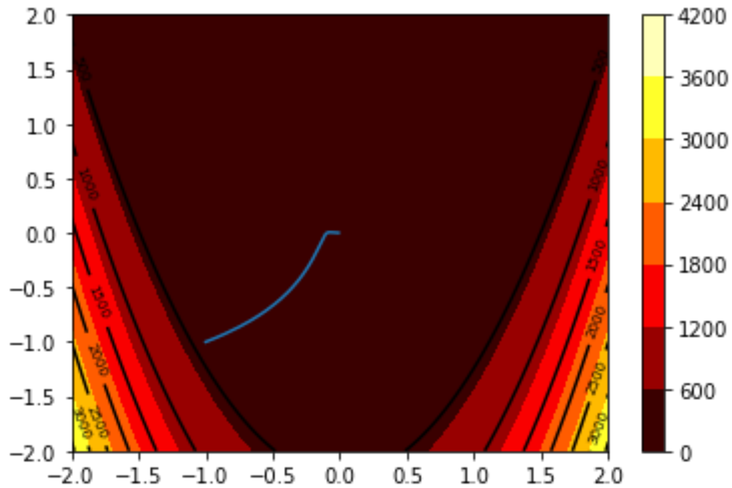


In [2]:
```
##5
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import cm

fig = plt.figure()
ax = fig.add_subplot(111)
u = np.linspace(-100, 100, 100)
x, y = np.meshgrid(u, u)
z = 100*(y-x**2)**2+(1-x)**2
cset = plt.contourf(x,y,z,6, cmap = plt.cm.hot)
contour = plt.contour(x,y,z,8, colors = 'k')
plt.clabel(contour, fontsize = 7, colors = 'k')
plt.colorbar(cset)
plt.show()
```



In [4]:
```
###the direction of arrow
import math
import numpy as np
start_point = np.array([-1,-1])
alpha = 0.0001
def z_value(x,y):
    z = 100*(y-x**2)**2+(1-x)**2
    return z

def f_prime(x,y):
    x_prime = -400*x*y+400*x**3+2*x-2
    y_prime = 200*(y-x**2)
    return np.array([x_prime,y_prime])
def point_update(x,y,alpha):
    grad = f_prime(x,y)
    grad_x = grad[0]
    grad_y = grad[1]
    x_new = x-alpha*grad_x
    y_new = y-alpha*grad_y
    return np.array([x_new,y_new])

point_update(start_point[0],start_point[1],alpha)
def gradient_decent(start_point, alpha, precision = 1, max_iters = 10000):
    z_list = []
    ite_list = [0]
    fig = plt.figure()
    x1_list = []
    x2_list = []
    ax = fig.add_subplot(111)
    u = np.linspace(-2, 2, 100)
    x, y = np.meshgrid(u, u)
    z = 100*(y-x**2)**2+(1-x)**2

    cset = plt.contourf(x,y,z,6, cmap = plt.cm.hot)
    contour = plt.contour(x,y,z,8, colors = 'k')
    plt.clabel(contour, fontsize = 7, colors = 'k')
    plt.colorbar(cset)
    previous = start_point
    update = point_update(previous[0],previous[1],alpha)
    x1_list.append(previous[0])
    x2_list.append(previous[1])
    ite = 1
    z_list.append(z_value(previous[0],previous[1]))

    ite_list.append(ite)
    while np.linalg.norm(f_prime(update[0],update[1]))>precision and np.lin
        previous = update
        update = point_update(previous[0],previous[1],alpha)
        ite +=1
        x1_list.append(previous[0])
        x2_list.append(previous[1])
        ite_list.append(ite)
        z_list.append(z_value(previous[0],previous[1]))
    z_list.append(z_value(update[0],update[1]))
    x1_list.append(update[0])
    x2_list.append(update[1])
```

```
    ax.plot(x1_list, x2_list)
    plt.show()
    return update, z_list, ite_list
z, y, x = gradient_decent(start_point, 0.0001)
print("end point: ", z)
```



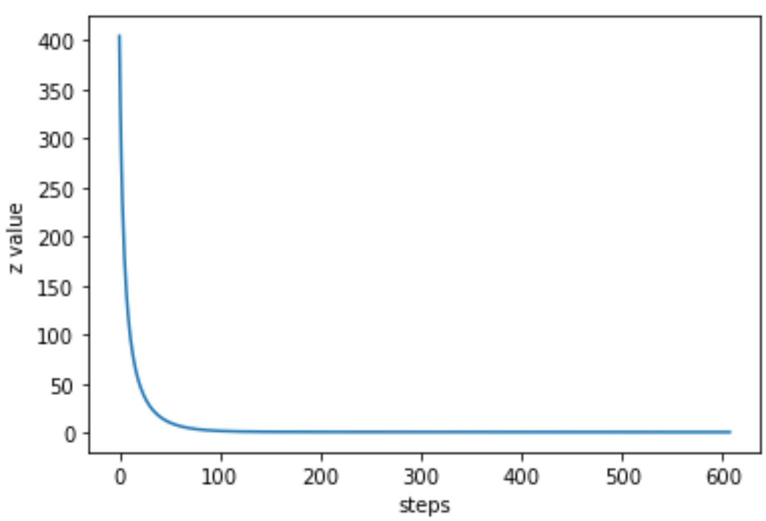
```
end point:  [-0.00997581  0.00050599]
```

In [5]:

```
fig = plt.figure()
plt.plot(y)
plt.xlabel("steps")
plt.ylabel("z value")
```

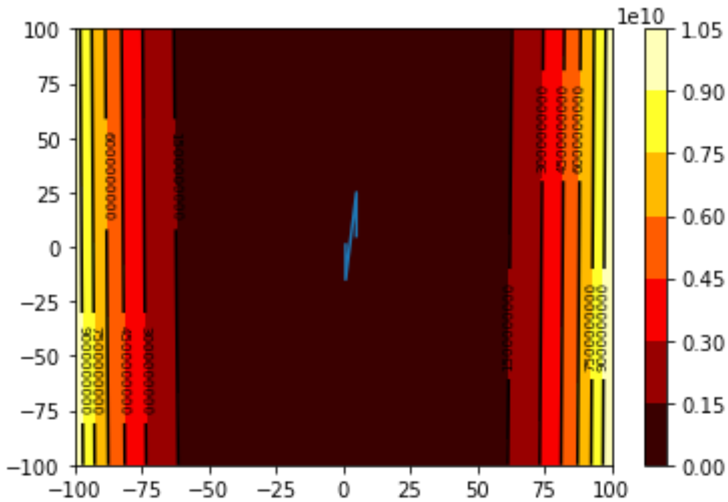
Out[5]:

```
Text(0, 0.5, 'z value')
```



In [6]:
```
import math
import numpy as np

def z_value(x,y):
    z = 100*(y-x**2)**2+(1-x)**2
    return z

def hess_inv(x,y):
    m = np.array([
        [-400*y+1200*x**2+2, -400*x],
        [-400*x,             200]
    ])
    return np.linalg.inv(m)
def f_prime(x,y):
    x_prime = -400*x*y+400*x**3+2*x-2
    y_prime = 200*(y-x**2)
    return np.array([x_prime,y_prime]).reshape(2,1)

def point_update_newton(x,y):
    grad = f_prime(x,y)
    point = np.array([x,y]).reshape(2,1)
    new = point-np.matmul(hess_inv(x,y),grad)
    return new

def Newtons_method(start_point, precision = 1, max_iters = 100):
    z_list = []
    ite_list = []
    fig = plt.figure()
    ax = fig.add_subplot(111)
    u = np.linspace(-100, 100, 100)
    x, y = np.meshgrid(u, u)
    z = 100*(y-x**2)**2+(1-x)**2
    x1_list = []
    x2_list = []
    cset = plt.contourf(x,y,z,6, cmap = plt.cm.hot)
    contour = plt.contour(x,y,z,8, colors = 'k')
    plt.clabel(contour, fontsize = 7, colors = 'k')
    plt.colorbar(cset)

    previous = start_point
    update = point_update_newton(previous[0][0],previous[1][0])
    x1_list.append(previous[0][0])
    x2_list.append(previous[1][0])

    ite = 0
    ite_list.append(ite)
    z_list.append(z_value(previous[0][0],previous[1][0]))

    while np.linalg.norm(f_prime(update[0][0],update[1][0]))>precision and
        previous = update
        update = point_update_newton(previous[0][0],previous[1][0])
        ite +=1
        ite_list.append(ite)
        z_list.append(z_value(previous[0][0],previous[1][0]))
        x1_list.append(previous[0][0])
        x2_list.append(previous[1][0])
```

```
    z_list.append(z_value(update[0][0],update[1][0]))
    x1_list.append(update[0][0])
    x2_list.append(update[1][0])
    ax.plot(x1_list, x2_list)
    plt.show()
    return update, z_list, ite_list
z, y, x = Newtons_method(np.array([5,5]).reshape(2,1))
print("end point: ", z)
```

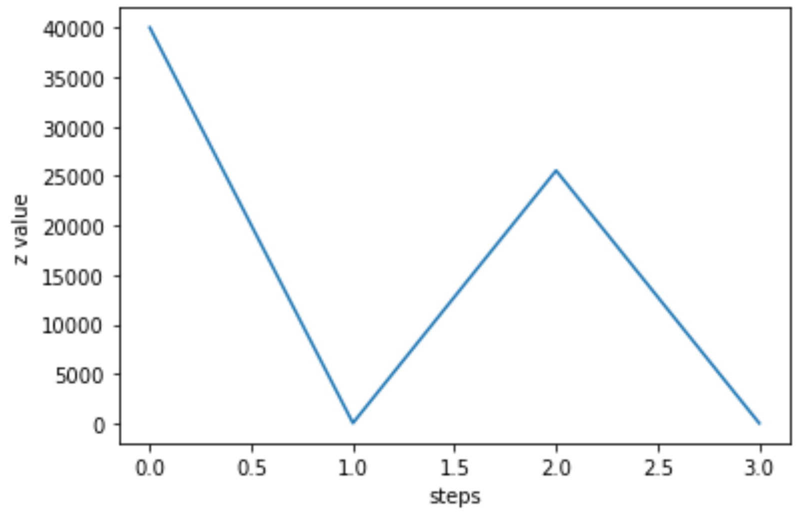


```
end point:  [[1.00079899]
 [1.00159862]]
```

In [7]:
```
fig = plt.figure()
plt.plot(y)
print(y)
plt.xlabel("steps")
plt.ylabel("z value")
```

```
[40016, 15.992002999100214, 25553.976901245092, 6.383860142168448e-07]
```

Out[7]:
```
Text(0, 0.5, 'z value')
```



In [ ]: