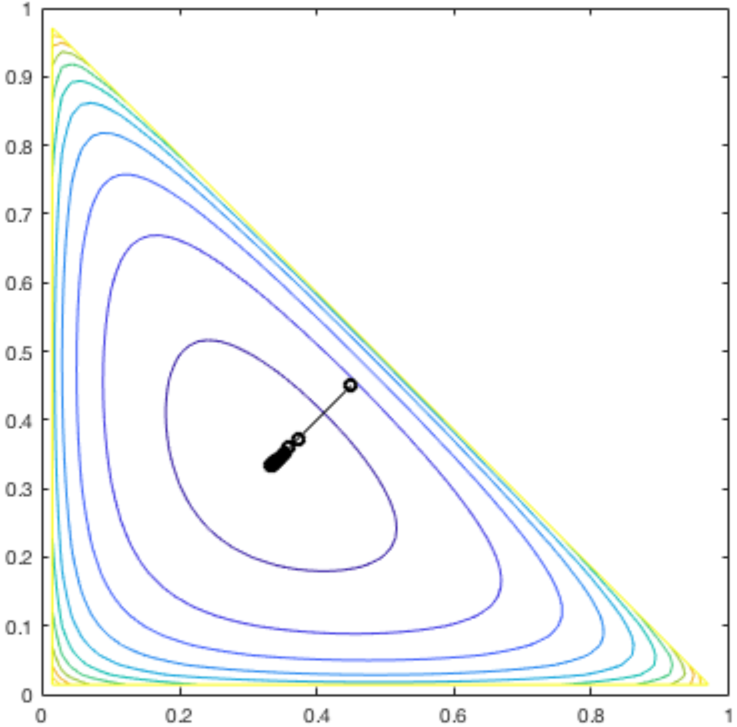
```
% Neural Network ECE 559 - Fall 2018
% Homework 3
% Montagna Marco
% exercise 2

close all
clear all
clc

% initial point
wx(1) = 0.45; % this belong to D
wy(1) = 0.45;
% initial point
wnx(1) = wx(1); % this belong to D
wny(1) = wy(1);

% maximum number of allowed iterations
max_iter = 1000;
% step size
eta = 0.01;
% define the function
syms x y;
f = -log(-x-y+1) - log(x) - log(y);

% gradient descend method
figure(1);
fsurf(f,[0 1 0 1])
hold on

figure(2);
clf; %clear figure
fcontour(f,[0 1 0 1]);
axis equal;
hold on

Z = -log(-wnx(1)-wny(1)+1) - log(wnx(1)) - log(wny(1));
 figure(5);
    plot(i,Z, '*')
    hold on
    xlabel('iteration number')
    ylabel('energy(w)')
    title('energy gradient method')

i = 1; % number of iterations

% % gradient computation
df_dx = diff(f, x);
df_dy = diff(f, y);
grad = [subs(df_dx,[x,y], [wx(1),wy(1)]) subs(df_dy, [x,y],
 [wx(1),wy(1)])];
% search direction, opposite to grad as we have seen in class
s = -(grad);
```

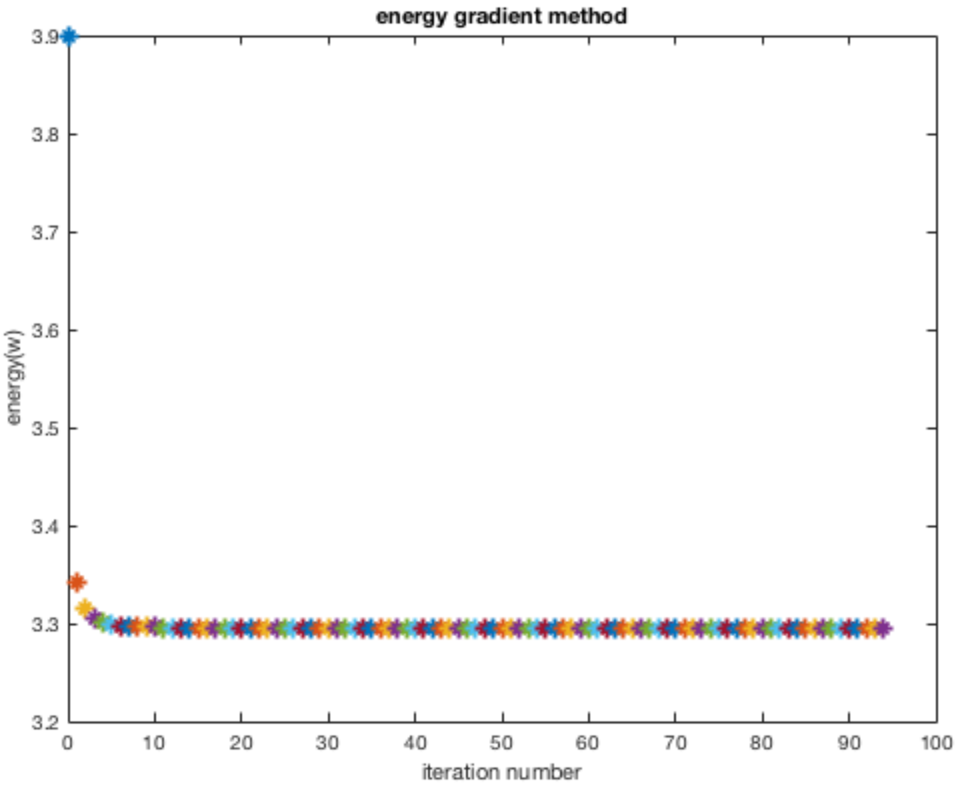
```
% gradient descend algorithm
while (abs(grad) > 0)
    w = [wx(i), wy(i)]';
    % update point
    wx(i+1) = wx(i)+eta*s(1);
    wy(i+1) = wy(i)+eta*s(2);
    % plot current point
    figure(2);
    plot([wx(i) wx(i+1)],[wy(i) wy(i+1)],'ko-')

    figure(1);
    Z = -log(-wx(i+1)-wy(i+1)+1) - log(wx(i+1)) - log(wy(i+1));
    scatter3(wx(i+1),wy(i+1),Z);

    figure(5);
    plot(i,Z, '*')



    i = i+1;
    grad = [subs(df_dx,[x,y], [wx(i),wy(i)]) subs(df_dy, [x,y],
 [wx(i),wy(i)])];
    s = -(grad);
end

% newton methon

k = 1; % number of iterations newton method
eta_n = 1;
d2f_dx2 = diff(df_dx, x);
d2f_dy2 = diff(df_dy, y);
d2f_dxdy = diff(df_dx, y);
d2f_dydx = d2f_dxdy;
% compute component hessian
H11 = subs(d2f_dx2,[x,y], [wnx(1),wny(1)]);
H12 = subs(d2f_dxdy, [x,y], [wnx(1),wny(1)]);
H21 = subs(d2f_dydx, [x,y], [wnx(1),wny(1)]);
H22 = subs(d2f_dy2, [x,y], [wnx(1),wny(1)]);
H =[H11 H12; H21 H22]; % hessian matrix
invH = inv(H);
grad = [subs(df_dx,[x,y], [wnx(1),wny(1)]) subs(df_dy, [x,y],
 [wnx(1),wny(1)])];
% search direction, opposite to grad as we have seen in class
s = -(grad);

figure(3);
clf; %clear figure
fcontour(f,[0 1 0 1]);
axis equal;
hold on

Z = -log(-wnx(1)-wny(1)+1) - log(wnx(1)) - log(wny(1));
```

```
figure(6);
    plot(k,Z, '*')
    hold on
    xlabel('iteration number')
    ylabel('energy(w)')
    title('energy newton method')

figure(4);
fsurf(f,[0 1 0 1])
hold on
invH=inv(H);

while (abs(grad) > 1e-15)
    wn = [wnx(k), wny(k)]'; % save points
    % update points newton method
    wnx(k+1) = wnx(k)+eta_n*invH(1,:)*s';
    wny(k+1) = wny(k)+eta_n*invH(2,:)*s';
    % plot current point
    figure(3);
    plot([wnx(k) wnx(k+1)],[wny(k) wny(k+1)],'ko-')

    figure(4);
    Z = -log(-wnx(k+1)-wny(k+1)+1) - log(wnx(k+1)) - log(wny(k+1));
    scatter3(wnx(k+1),wny(k+1),Z);

    figure(6);
    plot(k,Z, '*')


    k = k+1;
    % update hessian
    H11 = subs(d2f_dx2,[x,y], [wnx(k),wny(k)]);
    H12 = subs(d2f_dxdy, [x,y], [wnx(k),wny(k)]);
    H21 = subs(d2f_dydx, [x,y], [wnx(k),wny(k)]);
    H22 = subs(d2f_dy2, [x,y], [wnx(k),wny(k)]);
    H =[H11 H12; H21 H22];
    invH = inv(H);
    % update gradient
    grad = [subs(df_dx,[x,y], [wnx(k),wny(k)]) subs(df_dy, [x,y],
 [wnx(k),wny(k)])];
    s = -(grad);
end
```

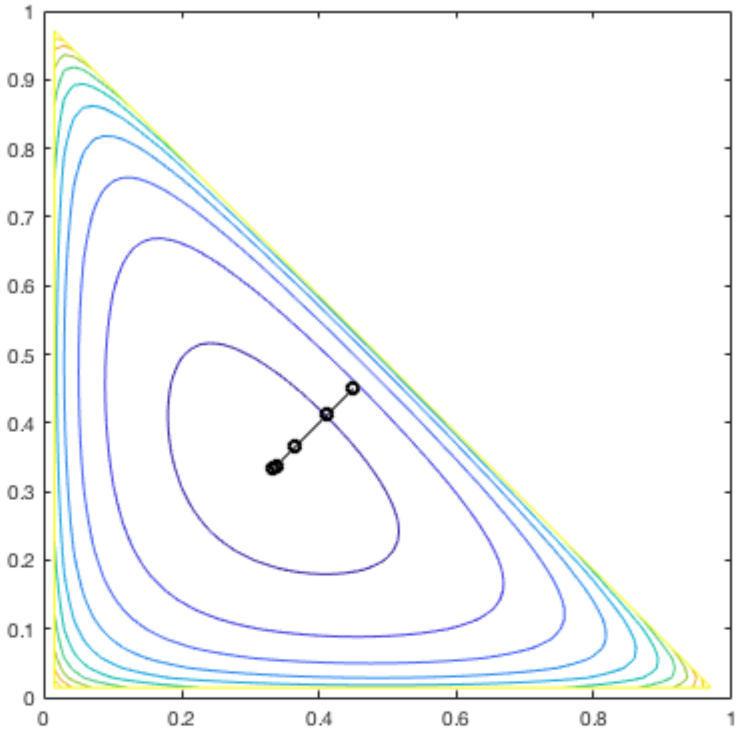
```
Warning: Imaginary parts of complex X and/or Y arguments ignored
```

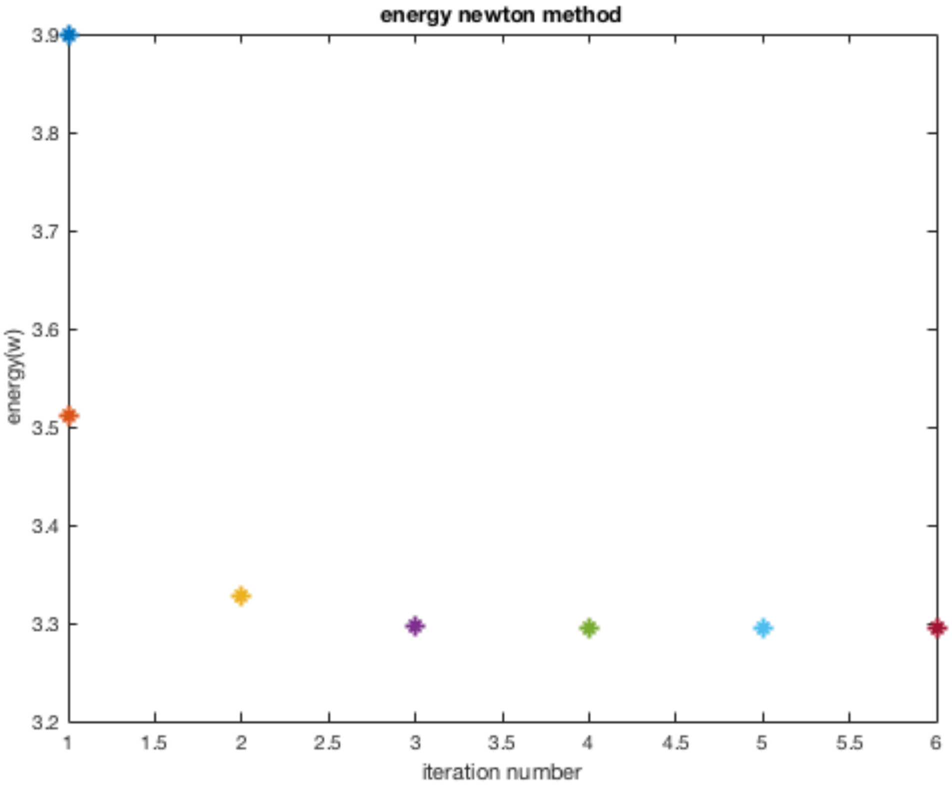
Gradient Descend Method





Newton’s Method




energy newton method
energy(w)
3.9
3.8
3.7
3.6
3.5
3.4
3.3
3.2
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
iteration number

Published with MATLAB® R2017b