MTE 203 - Advanced Calculus

Project 1
Extrema of Functions

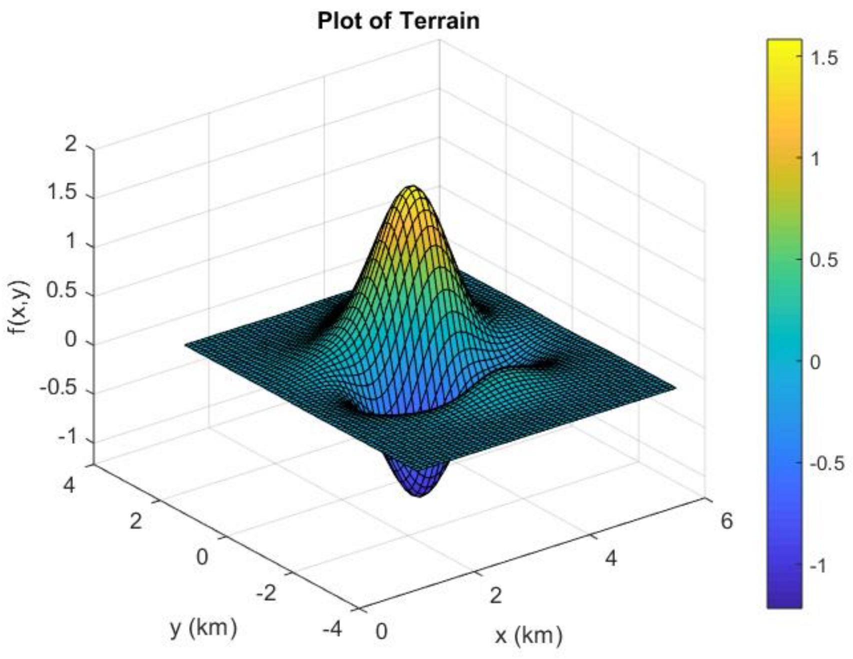
Name: Camille Walters
Student ID: 20716280

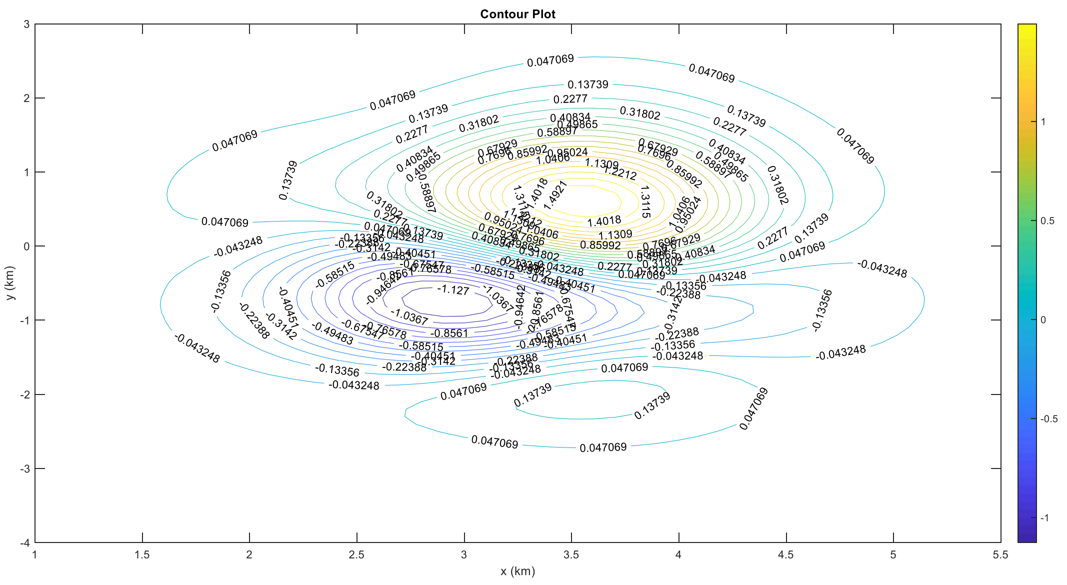
November 11, 2019

Section 1: Summary of Analysis

Part 1

a. This section uses the script 'onea.m'.To plot the surface, the function surf was used. The mesh of x and y were used across the desired range (1-5.5 for x and -4-3 for y) with an increment 0.1. To plot the contours, the function contour was used. 30 elevations were displayed, with labels. No calculations were required for this section. The plots can be seen below.





```
%using 0.1 unit increment
x=1:0.1:5.5;
y=-4:0.1:3;
[x y]=meshgrid(x,y);
%z function
z=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);


%plot of surface
figure(1)
surf(x,y,z);
xlabel('x (km)')
ylabel('y (km)')
zlabel('f(x,y)')
title('Plot of Terrain')
colorbar

%contour plot
figure(2)
[c h]=contour(x,y,z,30)
clabel(c,h)
xlabel('x (km)')
ylabel('y (km)')
title('Contour Plot')
colorbar
```

b. The area where the contour lines are the closest together show the steepest slope. This is because the rate of change as the function moves in the z-direction is faster than it moves in the x- or y-direction. This causes the lines of the contour plot to be close together since they represent changes in elevation. From the contour plot, this is at approximately (3.5, 0). This section refers to the script 'oneb.m'. To find the slope at a certain point, the gradient is calculated by taking the partial derivative with respect to x and y.

$$\nabla = \frac{\delta z}{\delta x}\hat{\imath} + \frac{\delta z}{\delta y}\hat{\jmath}$$

The point is then substituted in, then the magnitude of the vector is found by taking the square root of the sum of the squares of the components.

$$|\nabla| = \sqrt{\left(\frac{\delta z}{\delta x}\right)^2 + \left(\frac{\delta z}{\delta y}\right)^2}$$

To find the highest slope, the slope is calculated at every point within the domain at a step size of 0.05. This is accomplished by iterating through each point using nested for loops. The highest slope is retained as a variable (maxslope) as well as the coordinates of this slope (maxx and maxy). The slope at the point is compared to the current maximum slope, and if it is larger, its value replaces maxslope. Once every point has been checked, the maximum value is displayed.

```
syms x y;
z=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);

%partial derivatives
dx=diff(z,x)
dy=diff(z,y)
%magnitude of the gradient, which is the slope
gradient(x,y)=sqrt(dx.^2+dy.^2)

%initializing max values
maxslope=0;
maxx=0;
maxy=0;

%iterating through all values of x and y and checking
gradient value
%incrementing by value of 0.05 for precision
for i= 1:0.05:5.5
    for j= -4:0.05:3
        if gradient(i,j)>maxslope
            %reassigning values if gradient that was just
checked is larger
            maxslope=double(gradient(i,j));
            maxx=i;
            maxy=j;
        end
    end
end

disp(maxx)
disp(maxy)
```

c. First, the critical points needed to be found. This is found in the script 'criticalpoints.m'. The first partial derivatives of the function were displayed. These equations were then converted into forms that are compatible to use fsolve, meaning x was replaced with x(1) and y was replaced with x(2). These were declared as two functions (f1 and f2) in a function called "doublefunc". Using fsolve requires initial guesses, which were obtained from the contour plot found in part a. There were 3 critical points, so the function was used for 3 values to determine the exact values. The function was then evaluated at these points.

```
syms x y z
%displaying first derivative
z(x,y)=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);

disp(diff(z,x))
disp(diff(z,y))

%initial guesses based on contour plot
x0 = [3,-1];
x1 = [3.5,-2];
x2 = [3.75, 0.5];

xa = fsolve(@doublefunc,x0)
xb = fsolve(@doublefunc,x1)
xc = fsolve(@doublefunc,x2)

function F=doublefunc(x)

f1=(cos(2*x(1))*exp(- (x(1) - 3)^2 - x(2)^2/2)*(64*x(1)^2 +
16*x(1) + x(2)^2))/400 + (exp(- (x(1) - 3)^2 -
x(2)^2/2)*(128*x(1) + 16)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4)))/800 - (exp(- (x(1) - 3)^2 - x(2)^2/2)*(2*x(1) -
6)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 +
16*x(1) + x(2)^2))/800;
f2=(x(2)*exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4)))/400 - (x(2)*exp(- (x(1) -
3)^2 - x(2)^2/2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800 +
(cos((3*(x(2)/2 - 2)^2)/4)*exp(- (x(1) - 3)^2 -
x(2)^2/2)*((3*x(2))/8 - 3/2)*(64*x(1)^2 + 16*x(1) +
x(2)^2))/400;

F=[f1 f2];
end
```

d. The points needed to be classified. This is found in the script "onec.m". The A, B and C values were found by taking the second partial derivative of the function. D was then declared as B^2-AC. Based on the conditions of D and A to classify the points, a series of if statements were created. The values of A,B,C,D and the classification was evaluated for each critical point.

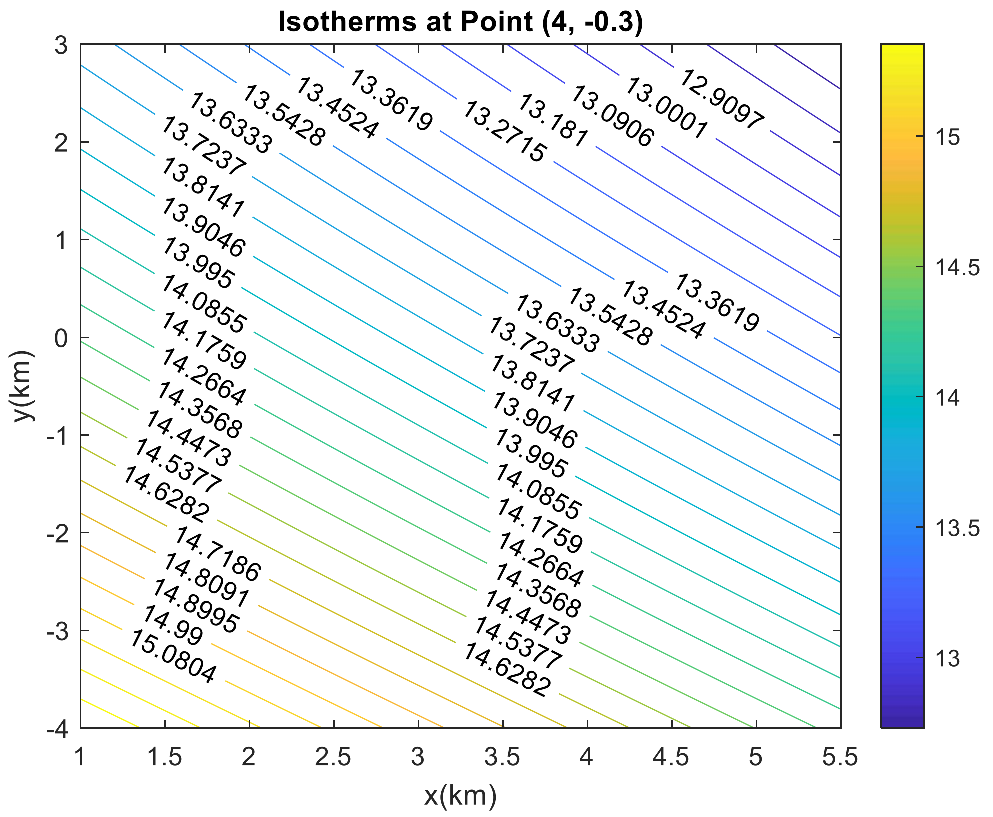
```
syms x y

%initializing parametrically to be able to sub in values
later
z(x,y)=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);

A(x,y)=diff(diff(z,x),x)
B(x,y)=diff(diff(z,x),y)
C(x,y)=diff(diff(z,y),y)
D(x,y)=B.^2-A.*C

%critical points (run with each uncommented)

% x=2.9191;
% y=-0.7505;

% x=3.5963;
% y=-2.0459;

x=3.5551;
y=0.6003;

solz=double(z(x,y))
sola=double(A(x,y))
solb=double(B(x,y))
solc=double(C(x,y))
sold=double(D(x,y))

if sold<0 && sola<0
    fprintf('relative max')
elseif sold<0 && sola>0
    fprintf('relative min')
elseif sold>0
    fprintf('saddle point')
elseif sold==0
    fprintf('no conclusion')
end
```

Part 2

a. This section can be found in the script 'twoa.m'. The temperature function was written in terms of symbolic variables so that the points can be easily substituted in. The maximum values found in part 1c) were then substituted in to find their values.

```
syms x y z

%initializing parametrically to be able to sub in values
t(x,y,z)=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-
1).^2-(z-1).^2))-10;

xmin=2.9191;
ymin=-0.7505;
zmin=-1.2236;
tempmin=double(t(xmin,ymin,zmin))

xmax=3.5551;
ymax=0.6003;
zmax=1.5883;
tempmax=double(t(xmax,ymax,zmax))

%used in part b. i.
xpoint=4;
ypoint=-0.3;
zpoint=0.1675;
temppoint=double(t(xpoint,ypoint,zpoint))
```

b. The point (4,-0.3) was substituted in to the function in part a to find its value. To find the isotherms, this value was used as the z-value in the temperature equation. Then, a contour plot was made with 30 contours across the given domain. This can be found in ‘isotherms.m’.



```
%using z value from part b. i.
z=0.1675;

x=1:0.1:5.5;
y=-4:0.1:3;
[x y]=meshgrid(x,y);
t=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-1).^2-(z-
1).^2))-10;

%use contour plot with a constant z value to find isotherms
[c h]=contour (x,y,t,30)
clabel (c,h)
xlabel('x(km)')
ylabel('y(km)')
title('Isotherms at Point (4, -0.3)')
colorbar
```

c. The solution to parts i and iii can be found in ‘twoc.m’. To find if the hiker is ascending or descending, the directional derivative of the terrain in the direction that the hiker is walking must be found. This is simply the dot product of the unit vector in the direction and the gradient at this point.

$$D_u f(x,y) = \nabla \cdot \hat{u}$$

If this value is positive, the hiker is ascending, and if the value is negative, the hiker is descending. First, the components of the gradient were found by taking the partial derivatives.

$$\nabla = \frac{\delta z}{\delta x}\hat{\imath} + \frac{\delta z}{\delta y}\hat{\jmath}$$

Then, the components of the unit vectors were expressed for both directions. The dot product was then found by evaluating the gradient at (4,-0.3) and summing the product of the components.

$$\nabla \cdot \hat{u} = \frac{\delta z}{\delta x}\hat{u}\,\hat{\imath} + \frac{\delta z}{\delta y}\hat{u}\hat{\jmath}$$

```
%this script solves c parts i. and iii.
syms x y z
z=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);
t=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-1).^2-(z-
1).^2))-10;

%components of the gradient
dx(x,y)=diff(z,x)
dy(x,y)=diff(z,y)

%northwest direction unit vector
i1=-1./sqrt(2);
j1=1./sqrt(2);

%southwest direction unit vector
i2=-1./sqrt(2);
j2=-1./sqrt(2);


%directional derivative of the function going north west
ddnw=double((i1.*dx(4,-0.3))+(j1.*dy(4,-0.3)))

%directional derivative of the function going south west
ddsw=double((i2.*dx(4,-0.3))+(j2.*dy(4,-0.3)))
```

The solutions to parts ii and iv can be found in ‘twocii.m’. To find the change in temperature the directional derivative was again found. To find the unit vector, first the tangent plane was found at this point. This is found by using the gradient at this point as the normal value and substituting the desired point into the standard equation for planes:

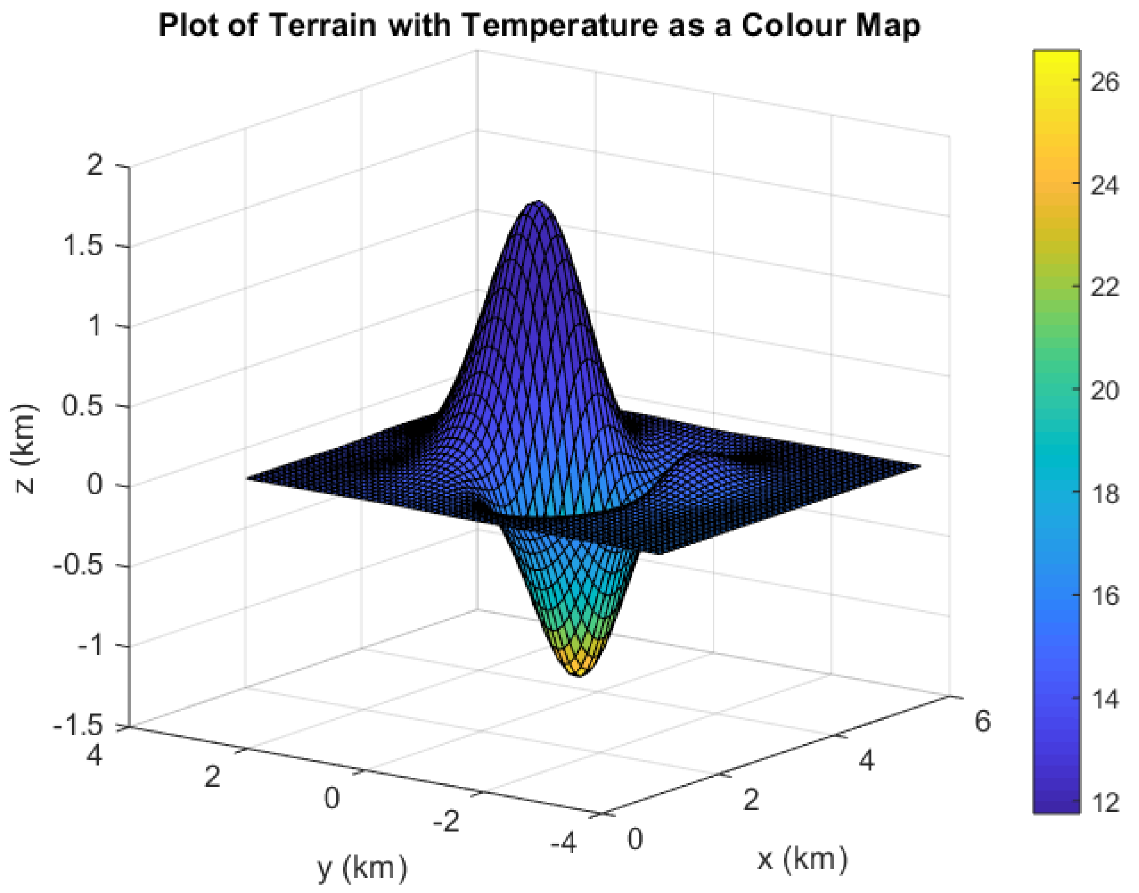
$$P(x,y) = \frac{\delta x}{\delta z}(x - x_0) + \frac{\delta y}{\delta z}(y - y_0)$$

This gave the z value for the unit vector at this point. The dot product was then again found between this unit vector and the gradient of the temperature function at this point.

```
syms x y z

t=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-1).^2-(z-
1).^2))-10;

%components of the gradient of the temperature function
dx(x,y,z)=diff(t,x)
dy(x,y,z)=diff(t,y)
dz(x,y,z)=diff(t,z)

z(x,y)=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);
elevation=z(4,-0.3);

%components of gradient of elevation function
dzdx(x,y)=diff(z,x)
dzdy(x,y)=diff(z,y)

tangentPlane(x,y)=dzdx(4,-0.3)*(x-4)+dzdy(4,-0.3)*(y+0.3);

%elevations at the points
NWP=tangentPlane(-1,-1);
SWP=tangentPlane(-1,1);

% to see the values of the points
disp(double(NWP))
disp(double(SWP))

%northwest direction unit vector
i1=-1./(sqrt(1+1+NWP.^2));
j1=1./(sqrt(1+1+NWP.^2));
k1=NWP./(sqrt(1+1+NWP.^2));

%southwest direction unit vector
i2=-1./(sqrt(1+1+SWP.^2));
j2=-1./(sqrt(1+1+SWP.^2));
k2=SWP./(sqrt(1+1+SWP.^2));
```

```
%directional derivative north west
ddnw=double((i1.*dx(4,-0.3,elevation))+(j1.*dy(4,-
0.3,elevation))+(j1.*dz(4,-0.3,elevation)))

%directional derivative south west
ddsw=double((i2.*dx(4,-0.3,elevation))+(j2.*dy(4,-
0.3,elevation))+(k2.*dz(4,-0.3,elevation)))
```

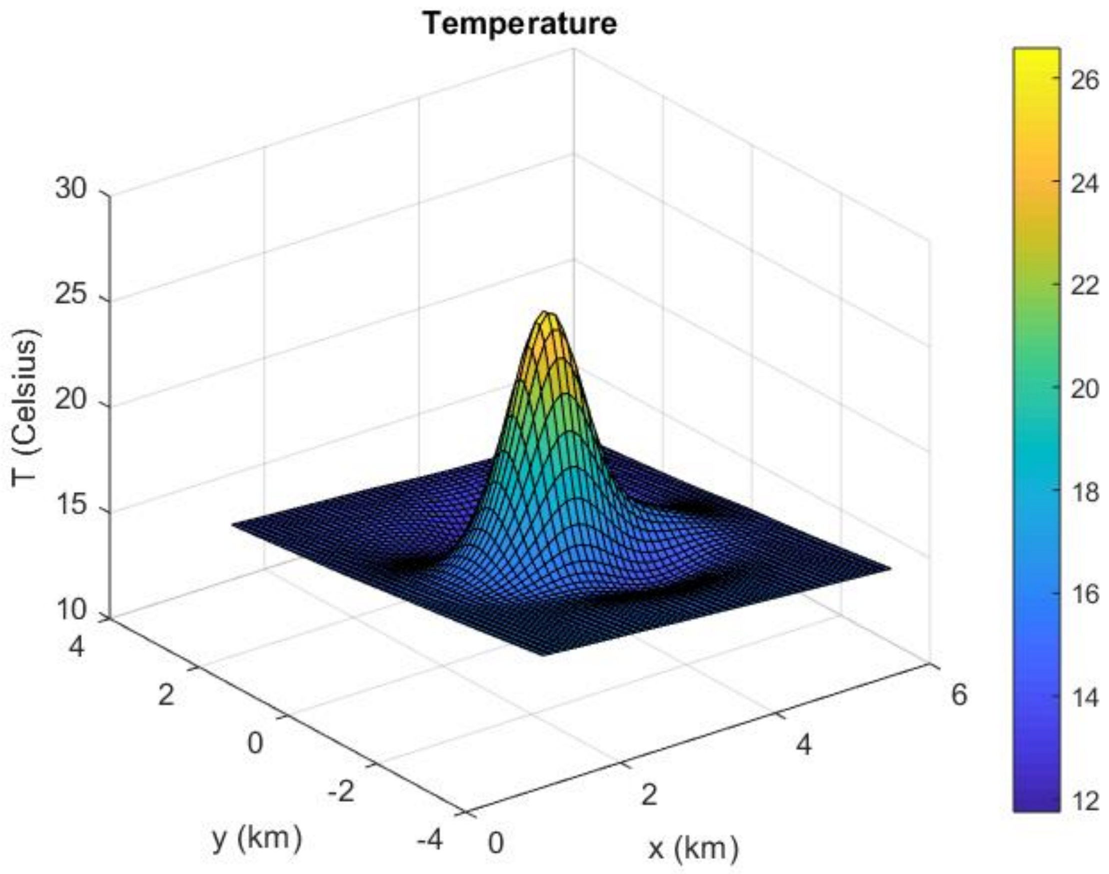
d. The solution can be found in ‘twod.m’. The elevation and temperature functions were declared. The plot was created using the surf function, with the temperature function as a fourth parameter. This is useful to the hiker since they will be able to predict how the temperature will increase or decrease as they move across the terrain. For example, it uses more physical exertion to climb a high elevation, and temperature increases also causes more physical exertion. The combination of these two factors allows the hiker to plan their route by determining the amount of physical exertion at certain areas.



```
x=1:0.1:5.5;
y=-4:0.1:3;
[x y]=meshgrid(x,y);
```

```
z=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);
t=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-1).^2-(z-
1).^2))-10;

surf(x,y,z,t);
title('Plot of Terrain with Temperature as a Colour Map')
xlabel('x (km)')
ylabel('y (km)')
zlabel('z (km)')
colorbar
```

e. The solution can be found in ‘twoe.m’. The elevation and temperature functions were declared, where the elevation is equal to z. Therefore, the z function will be substituted into the temperature function so that it is only in terms of x and y. This plot is useful to the hiker since they can see the temperature increases without the terrain plot. If they would like to visit a certain area on the terrain, they can easily see how the temperature will change at this area. This allows them to plan appropriately for the temperature.



```
x=1:0.1:5.5;
y=-4:0.1:3;
[x y]=meshgrid(x,y);

z=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2);
t=-0.1.*z.^2+17*exp(-0.1.*((0.1.*x-2)-(0.05.*y-1).^2-(z-
1).^2))-10;

surf(x,y,t)
title("Temperature")
xlabel('x (km)')
ylabel('y (km)')
zlabel('T (Celsius)')
colorbar
```

f. The equation to be optimized is the temperature function, and the constraint is the surface function. This can be found in “lagrange.m”. First, the constraint equation must be rewritten with z on the right side of the equation.

$$C = f(x, y) - z$$

Then, the Lagrange equation is written as the temperature plus the product of lambda and the constraint equation.

$$L = t + \lambda C$$

Next, the partial derivatives of this equation were found with respect to x, y, z and lambda. This gives four equations to be solved.

```
syms x y z lambda
%function to optimize
t=-0.1.*(0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2)).^2+17*exp(-0.1.*((0.1.*x-
2)-(0.05.*y-1).^2-((0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2))-1).^2))-10;
%constraint
C=0.00125.*exp(-((x-
3).^2+0.5.*y.^2)).*(sin(2.*x)+2.*sin(0.75*(0.5*y-
2).^2)).*(16.*x+64.*x.^2+y.^2)-z==0;
%lagrange equation
L=t+lambda.*lhs(C)
%partial derivatives
dx=diff(L,x)
dy=diff(L,y)
dz=diff(L,z)
dl=diff(L,lambda)
```

Solving these can be found in 'lagrangeSol.m'. The equations were rewritten with a change of variables, so that the function can be solved in terms of x. The x and y coordinates of the initial guess was based on the plot from part e. The z value was then solved at this point to give the z value initial guess. The initial guess for lambda was 0, since equation 3 is 0=-lambda. This gives the x,y and z coordinate. The temperature at this point was then solved.

```
%initial guess based on plot from part e
x0=[2.9 -0.8 -1.2163 0]
[x, fval]=fsolve(@solution, x0)


function F=solution (x)
%conversion of variables as follows

%x(1)=x
%x(2)=y
%x(3)=z
%x(4)=lambda
```

```
%functions from "lagrange.m"
f1=x(4)*((cos(2*x(1))*exp(- (x(1) - 3)^2 -
x(2)^2/2)*(64*x(1)^2 + 16*x(1) + x(2)^2))/400 + (exp(-
(x(1) - 3)^2 - x(2)^2/2)*(128*x(1) + 16)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4)))/800 - (exp(- (x(1) - 3)^2 -
x(2)^2/2)*(2*x(1) - 6)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800) +
17*exp(((exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2))/800 - 1)^2/10 - x(1)/100 + (x(2)/20 - 1)^2/10 +
1/5)*((((exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2))/800 - 1)*((cos(2*x(1))*exp(- (x(1) - 3)^2 -
x(2)^2/2)*(64*x(1)^2 + 16*x(1) + x(2)^2))/400 + (exp(-
(x(1) - 3)^2 - x(2)^2/2)*(128*x(1) + 16)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4)))/800 - (exp(- (x(1) - 3)^2 -
x(2)^2/2)*(2*x(1) - 6)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800))/5 - 1/100)
- (exp(- 2*(x(1) - 3)^2 - x(2)^2)*(128*x(1) +
16)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))^2*(64*x(1)^2 + 16*x(1) + x(2)^2))/3200000 -
(cos(2*x(1))*exp(- 2*(x(1) - 3)^2 - x(2)^2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2)^2)/1600000 + (exp(- 2*(x(1) - 3)^2 -
x(2)^2)*(4*x(1) - 12)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))^2*(64*x(1)^2 + 16*x(1) + x(2)^2)^2)/6400000

f2=17*exp(((exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x (1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2))/800 - 1)^2/10 - x(1)/100 + (x(2)/20 - 1)^2/10 +
1/5)*(x(2)/2000 + (((exp(- (x(1) - 3)^2 -
x(2)^2/2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800 -
1)*((x(2)*exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4)))/400 - (x(2)*exp(- (x(1) -
3)^2 - x(2)^2/2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800 +
(cos((3*(x(2)/2 - 2)^2)/4)*exp(- (x(1) - 3)^2 -
x(2)^2/2)*((3*x(2))/8 - 3/2)*(64*x(1)^2 + 16*x(1) +
x(2)^2))/400))/5 - 1/100) + x(4)*((x(2)*exp(- (x(1) - 3)^2
- x(2)^2/2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4)))/400 - (x(2)*exp(- (x(1) - 3)^2 -
x(2)^2/2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))*(64*x(1)^2 + 16*x(1) + x(2)^2))/800 +
(cos((3*(x(2)/2 - 2)^2)/4)*exp(- (x(1) - 3)^2 -
x(2)^2/2)*((3*x(2))/8 - 3/2)*(64*x(1)^2 + 16*x(1) +
x(2)^2))/400) - (x(2)*exp(- 2*(x(1) - 3)^2 -
x(2)^2)*(sin(2*x(1)) + 2*sin((3*(x(2)/2 -
2)^2)/4))^2*(64*x(1)^2 + 16*x(1) + x(2)^2))/1600000 +
(x(2)*exp(- 2*(x(1) - 3)^2 - x(2)^2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))^2*(64*x(1)^2 + 16*x(1) +
x(2)^2)^2)/3200000 - (cos((3*(x(2)/2 - 2)^2)/4)*exp(-
2*(x(1) - 3)^2 - x(2)^2)*((3*x(2))/8 - 3/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2)^2)/1600000

f3=-x(4)

f4=(exp(- (x(1) - 3)^2 - x(2)^2/2)*(sin(2*x(1)) +
2*sin((3*(x(2)/2 - 2)^2)/4))*(64*x(1)^2 + 16*x(1) +
x(2)^2))/800 - x(3)

F=[f1 f2 f3 f4]
end
```

Section 2: Summary of Results

1b) The location with the maximum slope is at (3.25,-0.1) with a slope of 3.1208.

1c)

Parameter	**Equation**
$\frac{\delta z}{\delta x}$	(cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800
$\frac{\delta z}{\delta y}$	(y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400
Initial Guesses	[x y]=[3,-1] , [3.5,-2], [3.75,0.5]
$A = \frac{\delta^2 z}{\delta x^2}$	(4*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/25 + (cos(2*x)*exp(- (x -

	3)^2 - y^2/2)*(128*x + 16))/200 - (sin(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/200 - (exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)^2*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - (cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(64*x^2 + 16*x + y^2))/200 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400
$B=\frac{\delta^2 z}{\delta x \delta y}$	(y*cos(2*x)*exp(- (x - 3)^2 - y^2/2))/200 - (y*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*((3*y)/8 - 3/2))/400 - (y*cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 - (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400 + (y*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800
$C=\frac{\delta^2 z}{\delta y^2}$	(exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y^2*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/200 + (3*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/3200 - (exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - (exp(- (x - 3)^2 - y^2/2)*sin((3*(y/2 - 2)^2)/4)*((3*y)/8 - 3/2)^2*(64*x^2 + 16*x + y^2))/400 + (y^2*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (y*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2))/100 - (y*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/200
D=B^2-AC	((y*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*cos(2*x)*exp(- (x - 3)^2 - y^2/2))/200 + (y*exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*((3*y)/8 - 3/2))/400 + (y*cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*((3*y)/8 -

	3/2)*(64*x^2 + 16*x + y^2))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800)^2 + ((sin(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/200 - (cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(128*x + 16))/200 - (4*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/25 + (exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/400 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)^2*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(64*x^2 + 16*x + y^2))/200 + (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400)*((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y^2*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/200 + (3*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/3200 - (exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - (exp(- (x - 3)^2 - y^2/2)*sin((3*(y/2 - 2)^2)/4)*((3*y)/8 - 3/2)^2*(64*x^2 + 16*x + y^2))/400 + (y^2*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (y*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2))/100 - (y*cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/200)

1d)

Point #	(x,y)	f(x,y)	A	B	C	D	Type of point
1	(2.9191, -0.7505)	-1.2236	4.4893	0.7479	4.8363	-21.1523	Relative minimum
2	(3.5963, -2.0459)	0.1796	-0.7658	0.2415	-1.4008	1.0145	Relative maximum
3	(3.5551, 0.6003)	1.5883	-5.8771	-0.5401	-4.2213	-24.5171	Relative maximum

All these points are appropriate, since they fall within the domain provided.

2a) The temperature at the highest elevation is 12.5386 degrees Celsius. The temperature at the lowest elevation is 26.6727 degrees Celsius.

2b)i The temperature at the point (4,-0.3) is 13.6988 degrees Celsius.

2c)i

Parameter	Equation
$\frac{\delta x}{\delta x}$	(cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800
$\frac{\delta x}{\delta y}$	(y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400
Gradient of elevation	$-0.3929\hat{\imath} + 1.5119\hat{\jmath}$
Unit vector	$\frac{-1}{\sqrt{2}}\hat{\imath} + \frac{1}{\sqrt{2}}\hat{\jmath}$

In the north west direction, the directional derivative is 1.3469, therefore he is ascending.

2c)ii

Parameter	Equation
$\frac{\delta t}{\delta x}$	-(17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5))/100
$\frac{\delta t}{\delta y}$	17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*(y/2000 - 1/100)
$\frac{\delta t}{\delta z}$	17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*(z/5 - 1/5) - z/5
Gradient of temperature	$-0.2370\hat{\imath} - 0.2406\hat{\jmath} - 3.9798\hat{k}$
Unit vector	$\frac{-1}{\sqrt{2 + 0.9064^2}}\hat{\imath} + \frac{1}{\sqrt{2 + 0.9064^2}}\hat{\jmath} + \frac{0.9064}{\sqrt{2 + 1.3469^2}}\hat{k}$

The temperature change that he experiences as he is walking is -2.3714 degrees Celsius/km.

2c)iii

Parameter	Equation
$\frac{\delta x}{\delta x}$	(cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) +

	2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800
$\frac{\delta x}{\delta y}$	(y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400
Gradient of elevation	$-0.3929\hat{\imath} + 1.5119\hat{\jmath}$
Unit vector	$\frac{-1}{\sqrt{2}}\hat{\imath} - \frac{1}{\sqrt{2}}\hat{\jmath}$

In the south west direction, the directional derivative is -0.7912, therefore he is descending.

2c)iv

Parameter	Equation
$\frac{\delta t}{\delta x}$	-(17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5))/100
$\frac{\delta t}{\delta y}$	17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*(y/2000 - 1/100)
$\frac{\delta t}{\delta z}$	17*exp((z - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*(z/5 - 1/5) - z/5
Gradient of temperature	$-0.2370\hat{\imath} - 0.2406\hat{\jmath} - 3.9798\hat{k}$
Unit vector	$\frac{-1}{\sqrt{2+3.9302^2}}\hat{\imath} - \frac{1}{\sqrt{2+3.9302^2}}\hat{\jmath} + \frac{3.9302}{\sqrt{2+3.9302^2}}\hat{k}$

The temperature change that he experiences as he is walking is -3.6303 degrees Celsius/km.

2f)

Parameter	Equation
Lagrange Equation (L)	17*exp(((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5) - lambda*(z - (exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800) - (exp(- 2*(x - 3)^2 - y^2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))^2*(64*x^2 + 16*x + y^2)^2)/6400000 - 10

$\frac{\delta L}{\delta x} = 0$	lambda*((cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800) + 17*exp(((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*((((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - 1)*((cos(2*x)*exp(- (x - 3)^2 - y^2/2)*(64*x^2 + 16*x + y^2))/400 + (exp(- (x - 3)^2 - y^2/2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/800 - (exp(- (x - 3)^2 - y^2/2)*(2*x - 6)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800))/5 - 1/100) - (exp(- 2*(x - 3)^2 - y^2)*(128*x + 16)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))^2*(64*x^2 + 16*x + y^2))/3200000 - (cos(2*x)*exp(- 2*(x - 3)^2 - y^2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2)^2)/1600000 + (exp(- 2*(x - 3)^2 - y^2)*(4*x - 12)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))^2*(64*x^2 + 16*x + y^2)^2)/6400000=0
$\frac{\delta L}{\delta y} = 0$	17*exp(((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - 1)^2/10 - x/100 + (y/20 - 1)^2/10 + 1/5)*(y/2000 + (((exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 - 1)*((y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400))/5 - 1/100) + lambda*((y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4)))/400 - (y*exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 + (cos((3*(y/2 - 2)^2)/4)*exp(- (x - 3)^2 - y^2/2)*((3*y)/8 - 3/2)*(64*x^2 + 16*x + y^2))/400) - (y*exp(- 2*(x - 3)^2 - y^2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))^2*(64*x^2 + 16*x + y^2))/1600000 + (y*exp(- 2*(x - 3)^2 - y^2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))^2*(64*x^2 + 16*x + y^2)^2)/3200000 - (cos((3*(y/2 - 2)^2)/4)*exp(- 2*(x - 3)^2 -

	y^2)*((3*y)/8 - 3/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2)^2)/1600000=0
$\frac{\delta L}{\delta y} = 0$	-lambda=0
$\frac{\delta L}{\delta \lambda} = 0$	(exp(- (x - 3)^2 - y^2/2)*(sin(2*x) + 2*sin((3*(y/2 - 2)^2)/4))*(64*x^2 + 16*x + y^2))/800 – z=0
Initial Guess (based on plot)	[x y z lambda]=[2.9 -0.8 -1.2163 0]

Using Lagrange Multipliers, the highest temperature is the point (2.9147, -0.7548, -1.2235) with a lambda value of 0. The temperature at this point is 26.6744 degrees Celsius.