Faculty of Engineering

EXTREMA OF FUNCTIONS

Project 1
MTE 203 - Advanced Calculus

Prepared by
Pavel Shering
ID #20523043
Instructor: Patricia Nieva

November 20, 2015

Table of Contents

List of Figures

List of Tables

Abstract

The following report demonstrates the use of MATLAB® mathematical software to solve a real life application of extreme value problem. The objective is to model the variation in temperature at the surface of a bounded mountainous terrain with a technique called Lagrange Multipliers, resulting in a system of non linear of equations. The resulting maximum temperature of the bounded region is 26.7°C. The MATLAB® allows to model complex functions and analyse complex multivariate problems.

1 Summary of Analysis

This section of the report describes the analysis and techniques followed to model the temperature distribution along the elevation of mountainous terrain described by Equation 1 and bounded by 2

$$z(x,y) = 0.00125e^{-((x-3)^2+(0.5)*y^2)}(sin(2*x) + 2sin(0.75(0.5y-2)^2)) \\ *(16x + 64x^2 + y^2) \tag{1}$$

$$R : \{1 < x < 5.5, -4 < y < 3\} \tag{2}$$

Elevation z is the distance from sea level described by xy-plane, where x & y are in km. The compass directions are identified as shown in Figure 1

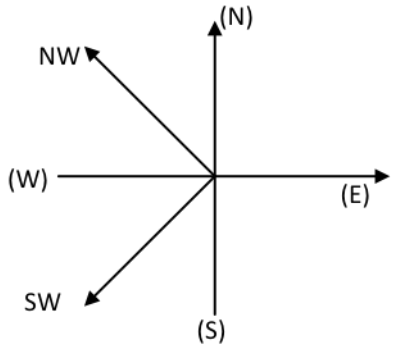

Figure 1: Directions of the Terrain

1.1 Part I

Figure 2 shows the surface of the island (mountainous terrain) where the color of the terrain is defined by the color-bar which refers to the elevation above sea level in km. This allows for spacial representation of the physical formation of the island made by using the surf() plot technique.

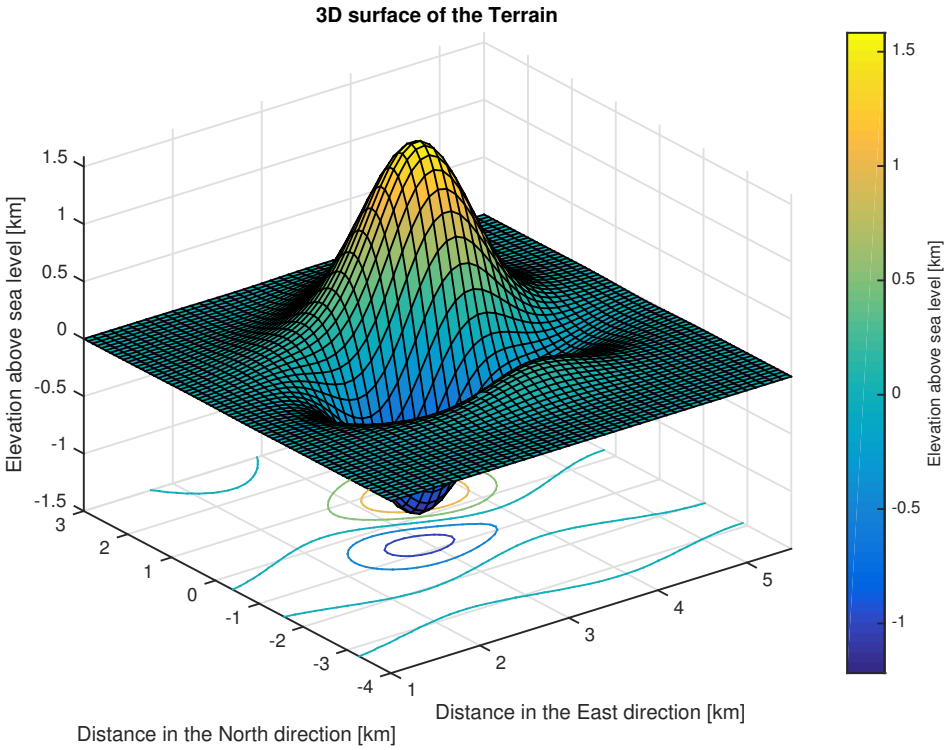

Figure 2: 3D surface plot of the Island - Mountainous Terrain

Figure 3 shows the contour map of the entire terrain which allows for 2D representation of the island and its elevation at approximately each coordinate. The accuracy of the elevation estimation can be increased by increasing the number of contour lines drawn as a parameter of the contour() function. However, by instruction the maximum amount of contours is displayed, valued at 30 lines.

Furthermore, by observation of Figure 3, it is determined where the slope of the terrain is the steepest. This is done by recognition of tightly packed contour lines, which by eye the steepest slope is interpolated to be at (3.3 , 0) coordinate.

However, verifying the guess mathematically is done by determining the general equation for gradient (Equation 3) of the surface at each coordinate on the surface plot, which represents the change in elevation, in other words slope of the terrain.

$$\nabla z(x,y) = \left(\frac{\partial z}{\partial x}\hat{i} + \frac{\partial z}{\partial y}\hat{j}\right) \tag{3}$$

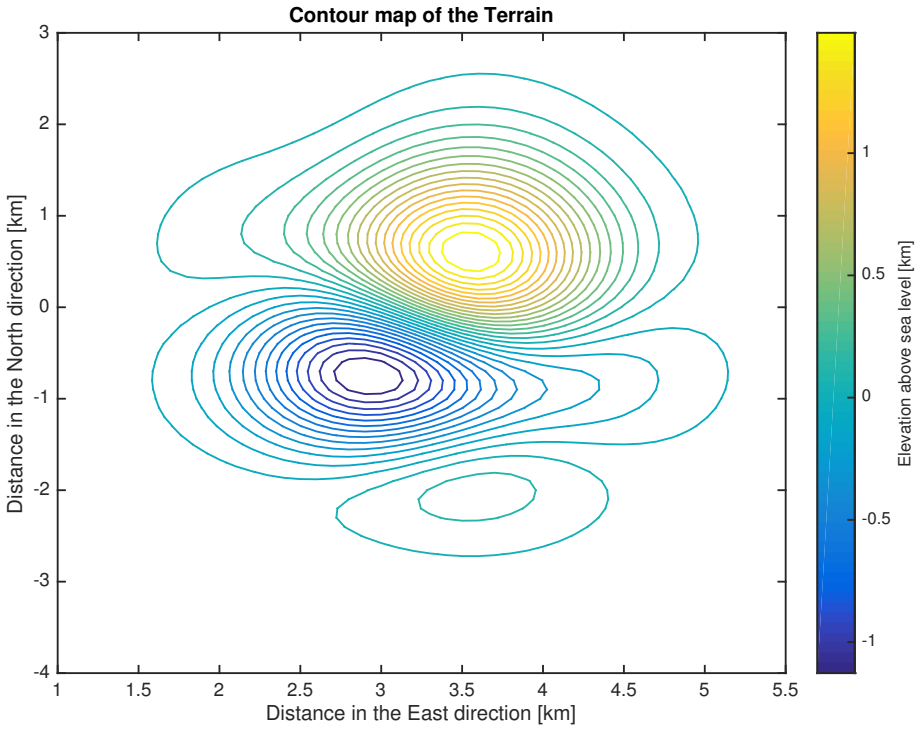

Figure 3: Contour Map of the Terrain

Additionally, to find the magnitude of the gradient at any point Equation 4 is used. To find the steepest (maximum) slope the gradient magnitude at each point of the surface plot must be compared against one another. This algorithm can be found in Appendix A.1 under Part Bi section of the MATLAB code, and the exact result is listed in Section 2.

$$|\nabla z(x,y)| = \sqrt{\left(\frac{\partial z}{\partial x}\right)^2 + \left(\frac{\partial z}{\partial y}\right)^2} \tag{4}$$

It is useful for a hiker to know the steepness of the terrain, for the hiker to plan the path to take, as well as which areas of the island to avoid or be aware of really high slopes.

The highest and lowest points of the terrain are also determined through the First and Second Derivative Tests. First Derivative test is setting the first derivative of z to zero to solve for x & y coordinates (critical points) at which the change in elevation is zero. Second Derivative test then allows for classification of each of the critical points as seen in Table 1.

Table 1: Critical Point Classification

#	(x,y)	f(x,y)	A (11)	B (11)	C (11)	D (14)	Classification
1	(2.9191, -0.7505)	-1.2236	4.4892	0.7478	4.8362	-21.1517	Abs Min
2	(3.5963, -2.0459)	0.1796	-0.7658	0.2414	-1.4006	-1.0143	Rel Max
3	(3.5551, 0.6003)	1.5883	-5.8770	-0.5400	-4.2212	-24.5164	Abs Max

Formulas for A, B, C, D can be found in Appendix B

Table 1 is useful for a hiker, as that allows the hiker to prepare exact equipment that might be required for hiking, climbing or diving. The algorithm for classifying the Critical Points can be found in Appendix A.2.

1.2 Part II

The maximum and minimum temperatures on the island are obtained by evaluating the temperature function *T(x,y,z)* described by Equation 5 at the critical points found in Table 1.

$$T(x, y, z) = -0.1z^2 + 17e^{-0.1((0.1x-2)-(0.05y-1)^2-(z-1)^2)} - 10 \tag{5}$$

Where x, y, z are in km and T is in °C and is bounded by the same range 2 as *z(x,y)*. Being informed of the environment temperature range is very useful to determine weather conditions and environmental factors.

At location (4, -0.3) the hiker will feel 13.7 °C, which can be obtained by evaluating Equation 5 at $x = 4$, $y = -0.3$, and $z = z(4, -0.3)$ from Equation 1. Figure 4 shows the isotherms at hikers current location as well as the temperature of the entire terrain at the hiker's current elevation of 0.17 km above sea level. As the isotherm plot displays, there is a change of about 3 °C across the terrain going from $\tilde{1}$5.5°C on the South-West corner to $\tilde{1}$2.8°C in the North-East corner.

If the hiker decides to walk in the North-West direction from the current location of (4, -0.3, 0.17), the hiker will be ascending. This is determined by calculating the rate of change of the slope of the terrain, gradient. The gradient is then evaluated at the current location, however to determine the rate of change in the North-West direction, a unit vector is required. That is determined by Equation 6.

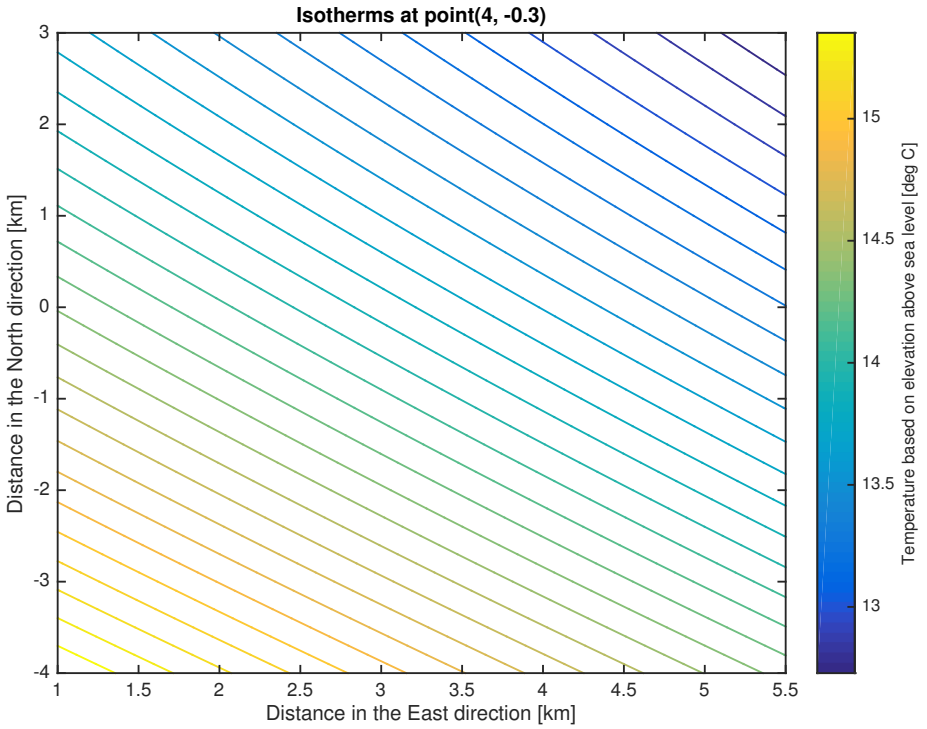

Figure 4: Isotherms at point(4, -0.3)

$$
\begin{aligned}
NW &= (-1, 1) \\
\hat{u}_{NW} &= \frac{(-1, 1)}{\sqrt{(-1)^2 + (1)^2}}
\end{aligned} \tag{6}
$$

The rate of change of slope in the North-West direction is then calculated by dot product of the gradient at current location and the unit vector in the direction of interest, shown in Equation 7, this is also know as directional gradient.

$$\nabla z(x, y)_{NW} = \nabla z_{(4,-0.3)} \cdot \hat{u}_{NW} \tag{7}$$

The hiker will also experience a change in temperature as steps are taken in the North-West direction. Therefore, directional gradient of T is required. However since T depends on x, y *&* z the change in T depends on change in elevation, z. The strategy is similar to determining the direction derivative of the slope, however the direction in North-West is shown in Equation 8 and the unit vector, respectively.

$$
\begin{aligned}
NW &= (-1, 1, \nabla z) \\
where\ \nabla z &= \delta x \left(\frac{\partial z}{\partial x}\right)_{(4,-0.3)} + \delta y \left(\frac{\partial z}{\partial y}\right)_{(4,-0.3)} \\
\hat{u}_{NW} &= \frac{(-1, 1, \nabla z)}{\sqrt{(-1)^2 + (1)^2 + (\nabla z)^2}}
\end{aligned}
\tag{8}
$$

Finally, the directional gradient for T is then found using Equation 9.

$$
\nabla T(x, y, z)_{NW} = \nabla T_{(4,-0.3,\nabla z)} \cdot \hat{u}_{NW} \tag{9}
$$

Similarly, if the hiker decides to travel in the South-West direction the process to determine the directional gradient of elevation will be the same as for North-West, however the direction vector changes to *SW = (-1 -1)* thereby also changing the unit vector $\hat{u}$. As for the directional gradient of temperature, ∇T [°C/km], in the South-West direction, the process is once again the same, besides the unit vector $\hat{u}$.

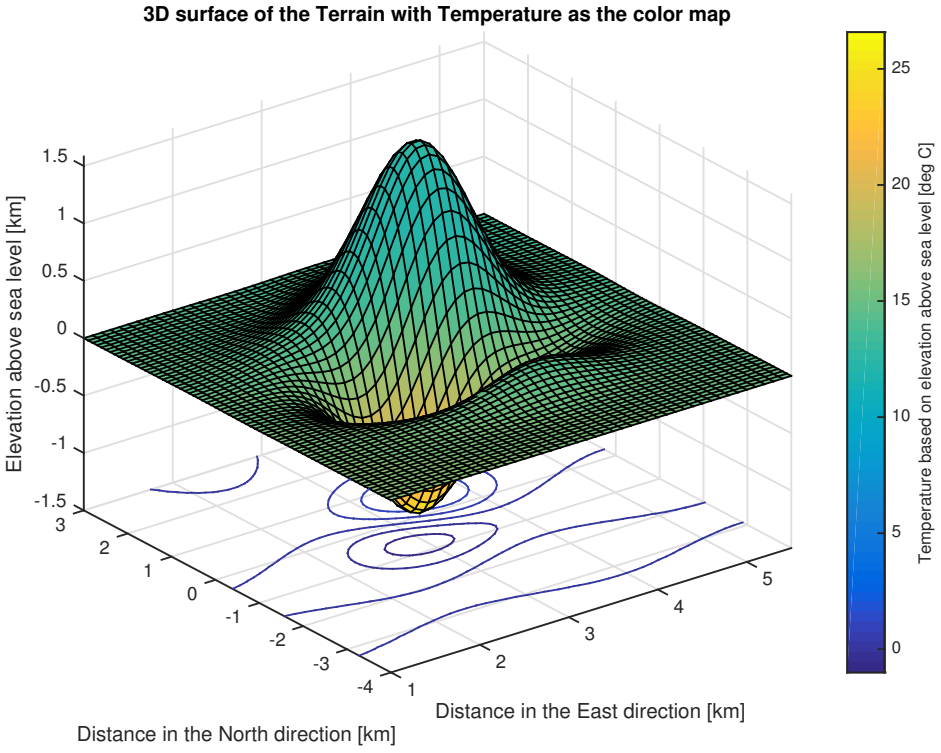

Figure 5: 3D Terrain Surface Temperature

The following Figure 5 and Figure 6 illustrate the temperature of the sur-

face of the terrain. This temperature color map can be very useful for hiker to understand the changes in temperature based on elevation, and the surface temperature trends of the island to prepare physically for the path the hiker decides to take. Hiker can notice that the temperature at sea level is approximately 15 °C, as well as the temperature increases as the elevation drops below sea level, and temperature decreases as the elevation goes above sea level.

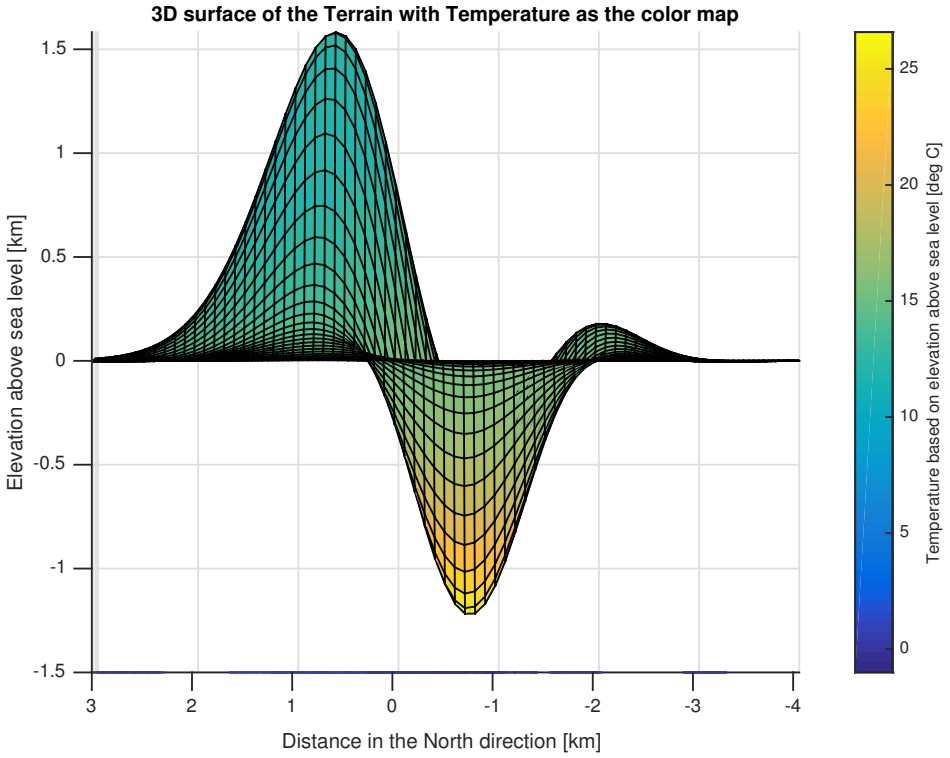

Figure 6: 3D Terrain Surface Temperature

Additionaly, Figure 7 is also very helpful to the hiker, as it provides the exact temperature on the surface at the exact coordinate. From this figure hiker can figure out the rate of change of temperature [°C/km] based on location changes, which can prepare the hiker for dramatic temperature drops based on the location change. This can even help the hiker to decide upon the path to take around the island.

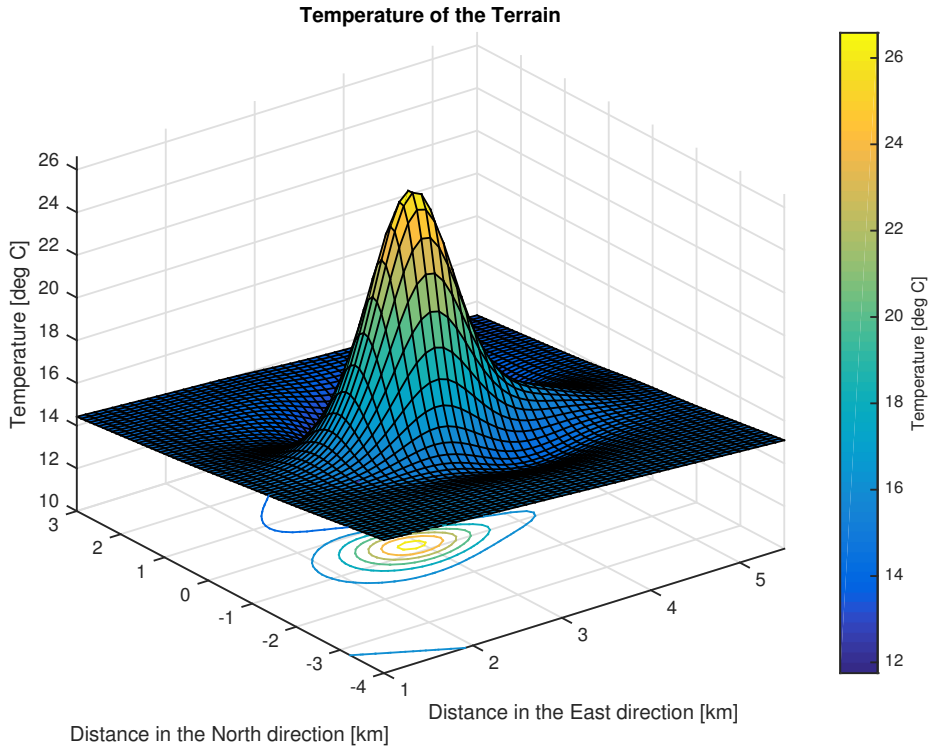

Figure 7: Temperature 3D surface plot

Finally, to determine the location of the highest temperature on the surface of the island, the Lagrange Multipliers technique is used which follows Equation 10.

$$L(x, y, z, \lambda) = T(x, y, z) + \lambda(z(x, y) - z) \tag{10}$$

L represents Lagrange Function, which is solved numerically for its roots, where $\lambda = 0$ condition cannot exist as the solution must be unique and not trivial. The gradient of the Lagrange function is set to zero to be solved numerically using vpasolve() with initial guesses for the (x,y,z) being the absolute minimum critical point since the Figure 2 was observed for maximum temperature, and λ = -20. Setting the gradient of the Lagrange function to zero, allows MATLAB to solve for *x, y, z,* & λ numerically, integrating from the initial guesses, since gradient of unconstrained ∇ *T(x,y,z)* = constrained λ ∇ *(z(x,y) - z)*. Therefore, solving for *(x,y,z)* is the location of highest temperature, and evaluating *T (x,y,z)* provides the value of maximum temperature on the island.

2 Summary of Results

2.1 Part I

Table 2: Steepest Slope

(x,y) of steepest slope	steepest slope z(x,y)
(3.2000, -0.1000)	3.1066

Table 3: Critical Point Classification

#	(x,y)	f(x,y)	A (11)	B (11)	C (11)	D (14)	Classification
1	(2.9191, -0.7505)	-1.2236	4.4892	0.7478	4.8362	-21.1517	Abs Min
2	(3.5963, -2.0459)	0.1796	-0.7658	0.2414	-1.4006	-1.0143	Rel Max
3	(3.5551, 0.6003)	1.5883	-5.8770	-0.5400	-4.2212	-24.5164	Abs Max

Formulas for A, B, C, D can be found in Appendix B. The Abs Min and Abs max were determined by looking at the f(x,y) value and comparing all the Critical Points

2.2 Part II

Table 4: Temperature at Extrema of Elevation

T(x,y,z) [°C]	(x,y)	Elevation z(x,y) [km]	Classification
26.6731	(2.9191, -0.7505)	-1.2236	Abs Min
12.5387	(3.5551, 0.6003)	1.5883	Abs Max

Table 5: Characteristics of Location (4, -0.3)

Location (x,y)	Elevation z(x,y) [km]	Temperature, T(x,y,z)[°C]
(4, -0.3)	0.1675	13.6987

Table 6: Gradient Changes due to Direction

Direction	Init (x,y)	∇ z(x,y) [km/km]	Asc/Des	∇ T(x,y,z) [°C/km]
(-1, 1) NW	(4, -0.3)	1.3469	Ascending	-3.1969
(-1, -1) SW	(4, -0.3)	-0.7912	Descending	2.7342

Table 7: Max Temperature on the Island

Location (x,y)	Elevation z(x,y) [km]	T(x,y,z) [°C]
(2.9147, -0.7547)	-1.2235	26.6747

Appendix A MATLAB Code

Appendix A.1 Main Script

```
% initialize matlab environment
close all;
clear all;
clc;
format short;

%% PART 1
% PART 1A
% creating the mesh for the plot with domain specified
    at 1 <= x <= 5.5 and -4 <= y <= 3 with fixed
    increment of 0.1 for deltaX and deltaY
[xnum,ynum] = meshgrid(1 : 0.1 : 5.5 , -4 : 0.1 : 3);

% surface equation in form of znum = f(xnum,ynum)
znum = @(xnum, ynum) 0.00125.*exp(-((xnum-3).^2+(0.5).*
    ynum.^2)).*(sin(2.*xnum)+2.*sin(0.75.*(0.5*ynum-2)
    .^2)).*(16.*xnum+64.*xnum.^2+ynum.^2);

% ploting the terrain as a 3D surface
figure
surfc(xnum,ynum,znum(xnum,ynum));
xlabel('Distance in the East direction [km]');
ylabel('Distance in the North direction [km]');
zlabel('Elevation above sea level [km]');
c = colorbar;
c.Label.String = 'Elevation above sea level [km]';
title('3D surface of the Terrain');

% ploting the contour map of the terrain over 30 levels
figure
[contour1, h]= contour(xnum,ynum,znum(xnum,ynum), 30);
xlabel('Distance in the East direction [km]');
```

```
ylabel('Distance in the North direction [km]');
zlabel('Height of the Terrain above sea level [km]');
c = colorbar;
c.Label.String = 'Elevation above sea level [km]';
%clabel(contour1, h);
title('Contour map of the Terrain');

% definition
syms x y z z_ l;
%surface equation in form of z = f(x,y)
z = symfun(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), [
    x y]);

% PART 1B
% taking first derivatives of the terrain function
zx(x,y) = diff(z,x);
zy(x,y) = diff(z,y);

% solving for gradient at every point in the bounded
    terrain
maxGrad = (gradient(z, [x,y]));
maxGrad_mag = 0;
for i = 1 : 0.1 : 5.5
    for j = -4 : 0.1 : 3
        new_maxGrad_mag = maxGrad(i,j);
        new_maxGrad_mag = sqrt(new_maxGrad_mag(1).^2 +
            new_maxGrad_mag(2).^2);
        if(new_maxGrad_mag > maxGrad_mag)
            maxGrad_mag = double(new_maxGrad_mag);
            indexMax = [i j];
        end
    end
end
disp(maxGrad_mag)
disp(indexMax)
```

```
61
62 % PART 1C
63 % define function handler
64 f = matlabFunction([zx zy]);
65 f = @(t) f(t(1), t(2));
66
67 % make initial guesses from the plotted graph
68 ip1 = [2.9, -0.8];
69 ip2 = [3.5, -2.0];
70 ip3 = [3.6,  1.1];
71
72 % FIRST DERIVATIVE TEST : solve for the critical points
73 cp1(1,:) = fsolve(f, ip1);
74 cp2(1,:) = fsolve(f, ip2);
75 cp3(1,:) = fsolve(f, ip3);
76
77 % displaying values of the terrain function at the
     Critical Points
78 zCP1 = double(z(cp1(1,1), cp1(1,2)));
79 zCP2 = double(z(cp2(1,1), cp2(1,2)));
80 zCP3 = double(z(cp3(1,1), cp3(1,2)));
81
82 % SECOND DERIVATIVE TEST : taking second derivatives of
     the terrain function
83 A(x,y) = diff(zx,x); % zxx(x,y)
84 C(x,y) = diff(zy,y); % zyy(x,y)
85 B(x,y) = diff(zx,y); % zxy(x,y)
86
87 % determine A B C for each point for the table in the
     report
88 double(A(cp1(1,1),cp1(1,2)));
89 double(B(cp1(1,1),cp1(1,2)));
90 double(C(cp1(1,1),cp1(1,2)));
91
92 double(A(cp2(1,1),cp2(1,2)));
93 double(B(cp2(1,1),cp2(1,2)));
```

```
94 double(C(cp2(1,1),cp2(1,2)));
95
96 double(A(cp3(1,1),cp3(1,2)));
97 double(B(cp3(1,1),cp3(1,2)));
98 double(C(cp3(1,1),cp3(1,2)));
99
100 % classify each Critical Point by finding the Hessian of
       the terrain z(x,y)
101 % D = B^2 - AC
102 D(x,y) = (B(x,y).^2) - A(x,y) * C(x,y);
103
104 % plugging in the CP points into the D equation
105 D1 = double(D(cp1(1,1),cp1(1,2)));
106 D2 = double(D(cp2(1,1),cp2(1,2)));
107 D3 = double(D(cp3(1,1),cp3(1,2)));
108
109 % classify the points based on D and A
110 classCritPoint(D1, A(cp1(1,1),cp1(1,2)), cp1);
111 classCritPoint(D2, A(cp2(1,1),cp1(1,2)), cp2);
112 classCritPoint(D3, A(cp3(1,1),cp1(1,2)), cp3);
113
114 %% PART 2
115
116 % Temperature distribution within the limits of the
       terrain is a function
117 % of the height and it is given by T(x,y,z) in [deg C],
       and x,y,z in km
118 Tnum = @(xnum, ynum) -0.1*(znum(xnum,ynum).^2) + 17*exp
       (-0.1*((0.1*xnum-2)-(0.05*ynum-1).^2-(znum(xnum,ynum)
       -1).^2))-10;
119 Tnum_3vars = @(xnum, ynum, znum) -0.1*(znum.^2) + 17*exp
       (-0.1*((0.1*xnum-2)-(0.05*ynum-1).^2-(znum-1).^2))
       -10;
120
121 T = symfun(-0.1*(z(x,y)^2) + 17*exp(-0.1*((0.1*x-2)
       -(0.05*y-1)^2-(z(x,y)-1)^2))-10, [x y]);
```

```
122 T_3vars = symfun(-0.1*(z_.^2) + 17*exp(-0.1*((0.1*x-2)
      -(0.05*y-1).^2-(z_-1).^2))-10, [x y z_]);
123
124 % PART 2A
125 % Calculating temperatures at highest and lowest
      elevations on the terrain
126 tLowElevation = double(T(cp1(1,1), cp1(1,2)))
127 tHighElevation = double(T(cp3(1,1), cp3(1,2)))
128
129 % PART 2B
130 % PART 2Bi
131 % Calculating temperature at point (4,-0.3)
132 P = [4  -0.3];
133 tempPoint = double(T(P(1,1), P(1,2)));
134
135 % PART 2Bii
136 height = znum(P(1,1), P(1,2))
137
138 figure;
139 contour(xnum, ynum, Tnum_3vars(xnum, ynum, height), 30);
140 xlabel('Distance in the East direction [km]');
141 ylabel('Distance in the North direction [km]');
142 zlabel('Height of the Terrain above sea level [km]');
143 c = colorbar;
144 c.Label.String = 'Temperature based on elevation above
      sea level [deg C]';
145 title('Isotherms at point(4, -0.3)');
146
147 %PART 2C
148 %PART 2Ci
149 u = [-1  1];
150 u = u / sqrt(u(1)^2 + u(2)^2);
151
152 % calculating gradient
153 grad(x,y) = [zx, zy];
154
```

```
155 % calculating the directional gradient at point P
      (4,-0.3)
156 dir_grad = double(dot(grad(P(1),P(2)),u))
157 if (dir_grad >0)
158     fprintf('Hiker traveling in Northwest direction is
          ascending');
159 else
160     fprintf('Hiker traveling in Northwest direction is
          descending');
161 end
162
163 % PART 2Cii
164 Tx(x,y,z_) = diff(T,x);
165 Ty(x,y,z_) = diff(T,y);
166 Tz(x,y,z_) = diff(T,z_);
167
168 gradT(x, y, z_) = [Tx, Ty, Tz];
169
170 changeZ = -1 * zx(P(1),P(2)) + 1 * zy(P(1),P(2));
171 u2 = [-1 1 changeZ]
172 u2 = u2 / sqrt(u2(1)^2 + u2(2)^2 +u2(3)^2)
173
174 % calculating the directional gradient
175 dir_grad2 = double(dot(gradT(P(1),P(2), changeZ),u2))
176
177 %PART 2D
178 %PART 2Di
179 u3 = [-1 -1];
180 u3 = u3 / sqrt(u3(1)^2 + u3(2)^2);
181
182 % calculating the directional gradient at point P
      (4,-0.3)
183 dir_grad3 = double(dot(grad(P(1),P(2)),u3))
184 if (dir_grad3 >0)
185     fprintf('Hiker traveling in SouthWest direction is
          ascending');
```

```
186 else
187     fprintf('Hiker traveling in SouthWest direction is
        descending');
188 end
189
190 % PART 2Dii
191 changeZ4 = -1 * zx(P(1),P(2)) + -1 * zy(P(1),P(2));
192 u4 = [-1 -1 changeZ4]
193 u4 = u4 / sqrt(u4(1)^2 + u4(2)^2 +u4(3)^2)
194
195 % calculating the directional gradient
196 dir_grad4 = double(dot(gradT(P(1),P(2), changeZ4),u4))
197
198 % PART 2E
199 % ploting the temperature as color of the surface
200 figure
201 surfc(xnum, ynum, znum(xnum,ynum), Tnum(xnum, ynum));
202 xlabel('Distance in the East direction [km]');
203 ylabel('Distance in the North direction [km]');
204 zlabel('Elevation above sea level [km]');
205 c4 = colorbar;
206 c4.Label.String = 'Temperature based on elevation above
      sea level [deg C]';
207 title('3D surface of the Terrain with Temperature as the
      color map');
208
209 % PART 2F
210 % ploting the temperature of the terrain as a 3D surface
211 figure
212 surfc(xnum,ynum,Tnum(xnum, ynum));
213 xlabel('Distance in the East direction [km]');
214 ylabel('Distance in the North direction [km]');
215 zlabel('Temperature [deg C]');
216 c = colorbar;
217 c.Label.String = 'Temperature [deg C]';
218 title('Temperature of the Terrain');
```

```
219
220 % PART 2G
221 L = T_3vars(x, y, z_) + l*(z(x, y) - z_)
222 Grad_L = gradient(L, [x y z_ l])
223 T_max_var = vpasolve(Grad_L == 0, [x, y, z_, l], [cp1(1)
        cp1(2) z(cp1(1), cp1(2)) -20])
224 T_max = T_3vars(T_max_var.x, T_max_var.y, T_max_var.z_)
```

Appendix A.2 Functions

```
1  function [ output_args ] = classCritPoint( D, A, cp)
2  %classPoint Summary of this function goes here
3  %   Classifying critical points using SECOND DERIVATIVE
       TEST
4  %   D < 0 and A < 0 then (x,y) is a REL MAX
5  %   D < 0 and A > 0 then (x,y) is a REL MIN
6  %   D > 0 then (x,y) is a SADDLE POINT
7  %   D == 0 then NON CONCLUSIVE
8
9      if (D < 0)
10         if (A < 0)
11             fprintf('(%f, %f) is a REALTIVE MAXIMUM.\n',
                   cp(1,1),cp(1,2));
12         else
13             fprintf('(%f, %f) is a REALTIVE MINIMUM.\n',
                   cp(1,1),cp(1,2));
14         end
15     elseif (D > 0)
16         fprintf('(%f, %f) is a SADDLE POINT.\n',cp(1,1),
               cp(1,2));
17     else
18         fprintf('(%f, %f) is UNCONCLUSIVE.\n',cp(1,1),cp
               (1,2));
19     end
20 end
```

Appendix B Calculations

$$A = \frac{\partial^2 z}{\partial x^2} \tag{11}$$

$$B = \frac{\partial^2 z}{\partial x \partial y} \tag{12}$$

$$C = \frac{\partial^2 z}{\partial y^2} \tag{13}$$

$$D = B^2 - AC \tag{14}$$