Finding An efficient Path For Mars Rover

Flowchart

Start

```
load Terrain.mat
V=0.01/1000;
cg=1;cycle=50;h=T/1000;C=1/20000;
[ segment,TerrainFile,xs,ys,xe,ye,nwp,wp ]
= getProjectInput( 'path.ini' );
```

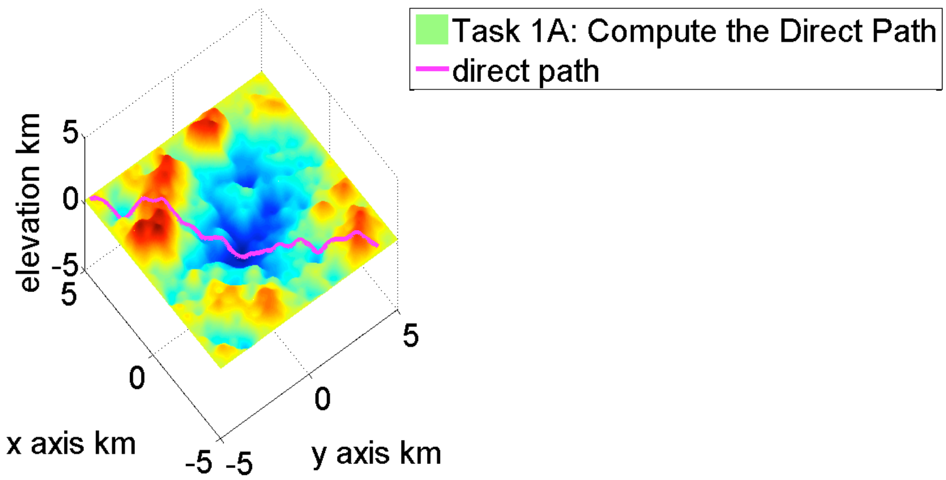
```
[xp_orig,yp_orig,hp_orig,d_orig,eplus_orig,eminus_orig]=computepath(xs,ys,xe,ye,nwp,wp
,x,y,h,segment);
hmin_orig=min(hp_orig);
hmax_orig=max(hp_orig);
[J_orig] = compute_cost( C,V,g,x,y,h,xs,xe,ys,ye,segment,nwp,wp );
[xp_opt              yp_opt              hp_opt              J_opt              eplus_opt
eminus_opt]=optimizewithcost(  x,y,h,dhdx,dhdy,g,C,V,cg,cycles,nwp,wp,xs,ys,xe,ye,segme
nt );
hmin_opt=min(hp_opt);
hmax_opt=max(hp_opt);
```

Output:
Metrics for the original path:
d_orig, J_orig, hmin_orig, hmax_orig, eplus_orig, eminus_orig
The original path:
xp_orig, yp_orig, hp_orig
Metrics for the optimized path:
d_opt, J_opt, hmin_opt, hmax_opt, eplus_opt, eminus_opt
The optimized path

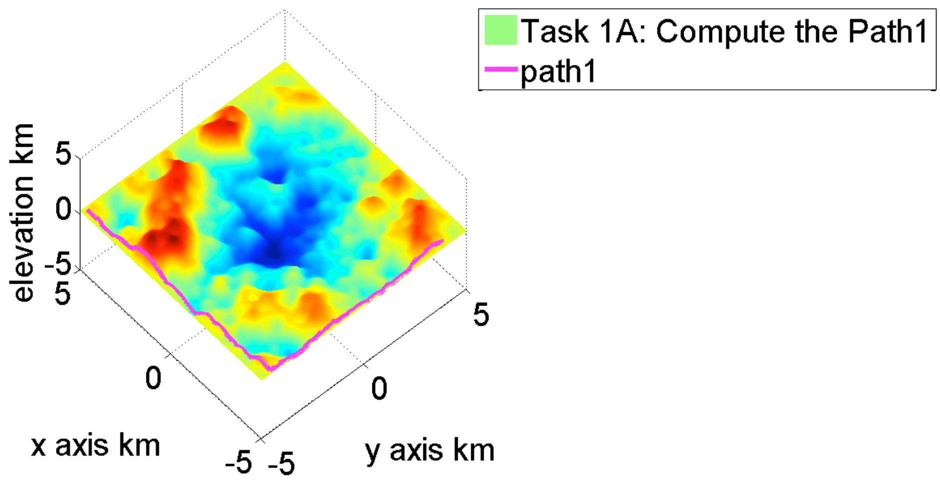
end

Task 1

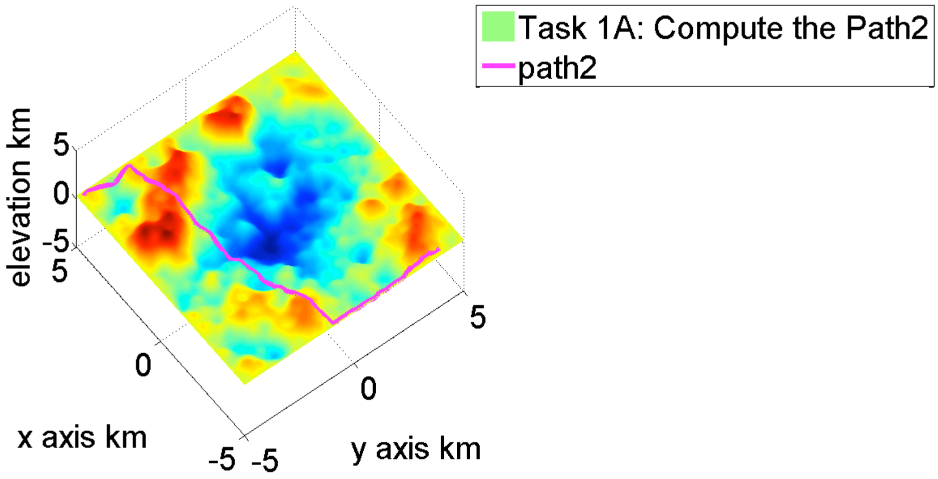
1.Include and succinctly discuss the tables and figures for parts A-E of Task 1. Make sure all the figures are have axis labels and legends where appropriate.



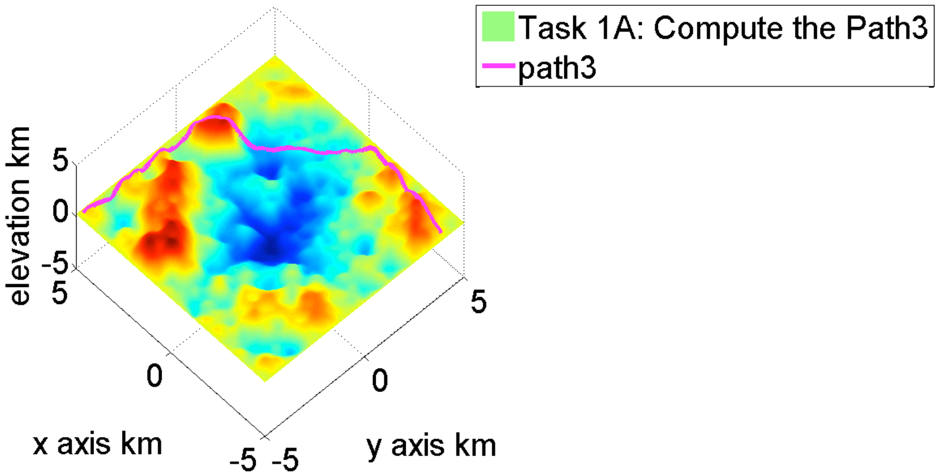
This figure describes the direct path with no way points.



This figure describes the path1 with 1 way point.



This figure describe the path2 for 2 way points.



This figure describes the path 3 with 2 way points.

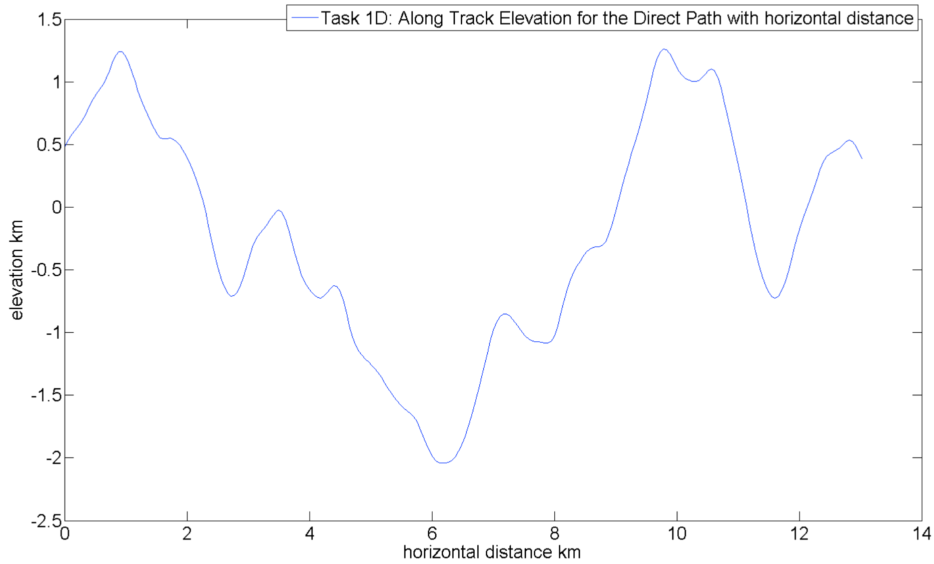
Task 1C: Comparison of 4 Paths

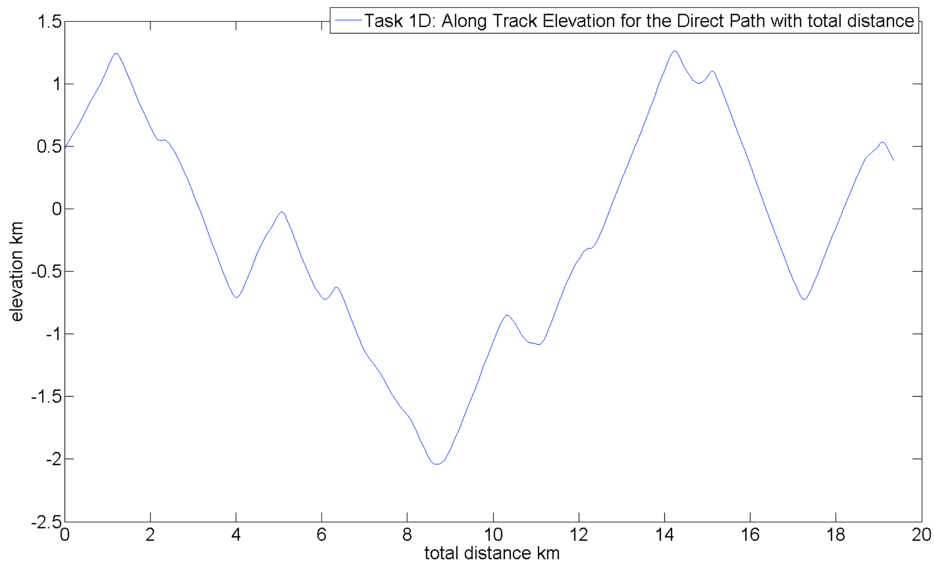
Route	Time(s)	Distance(km)	plusElev.(km)	minusElev.(km)
Direct	1.93E+06	17.84787666	5.473723422	5.559382255
Paht1	1.97E+06	19.67570946	2.974675037	3.06033387
Paht2	1.92E+06	19.16072202	3.852238329	3.937897162
Path3	2.11E+06	21.09318814	5.512815453	5.598474286

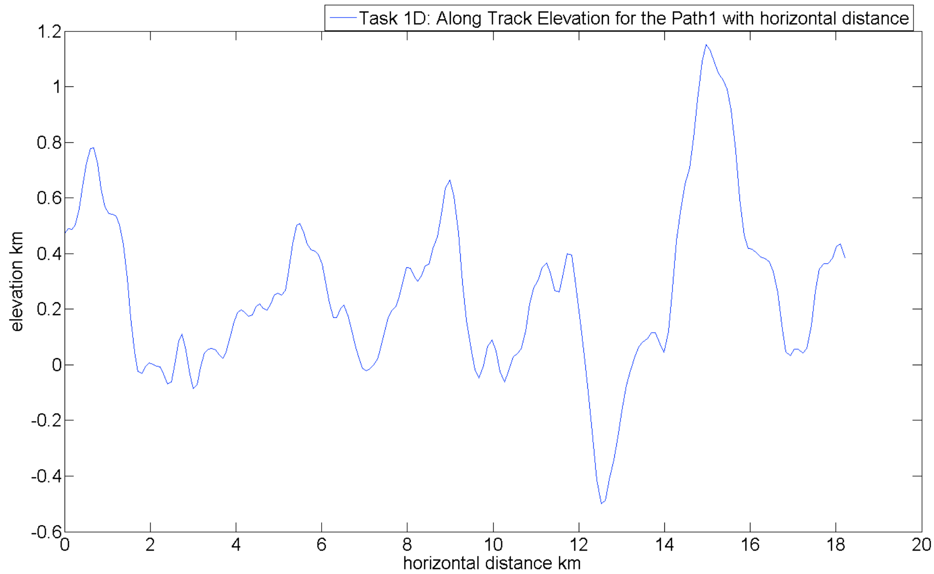
Path3 takes the longest time with longest distance with relatively big elevation difference.

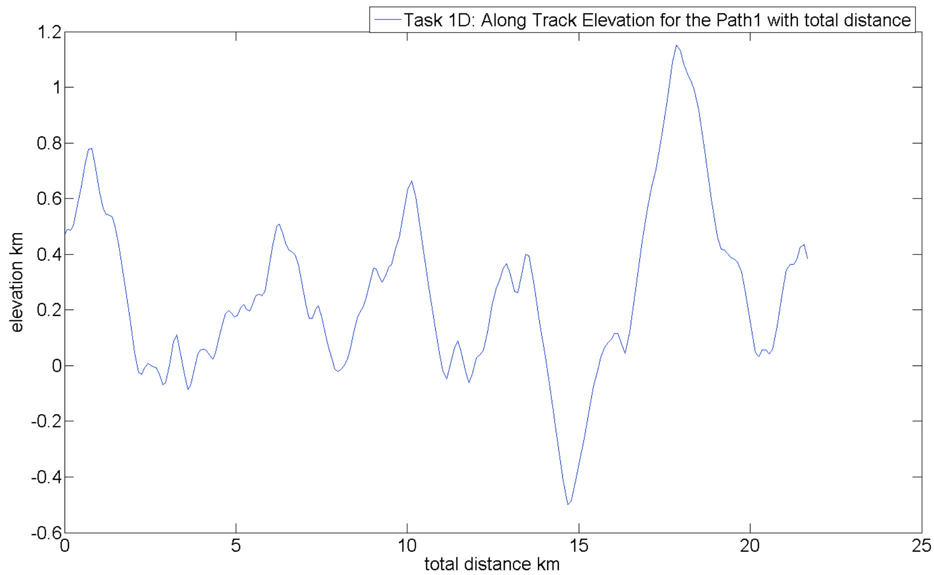
The direct path takes the shortest distance while takes long time and its elevation difference is huge.

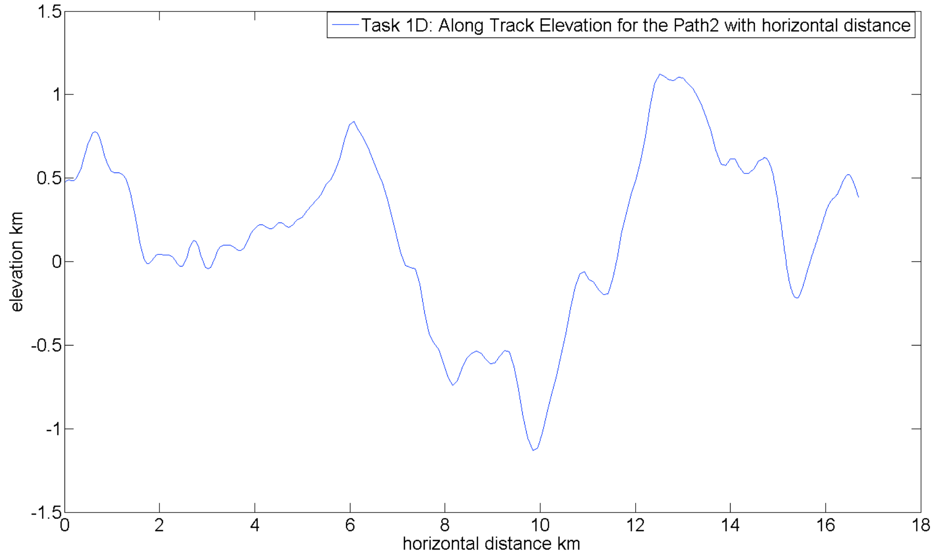
The path1 and path2 take almost the same time ,which is less than path 3 and the direct path, and distance but path1 does not have much path going uphill or downhill like path2.


Task 1D: Along Track Elevation for the Direct Path with horizontal distance
elevation km
horizontal distance km
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
2
4
6
8
10
12
14

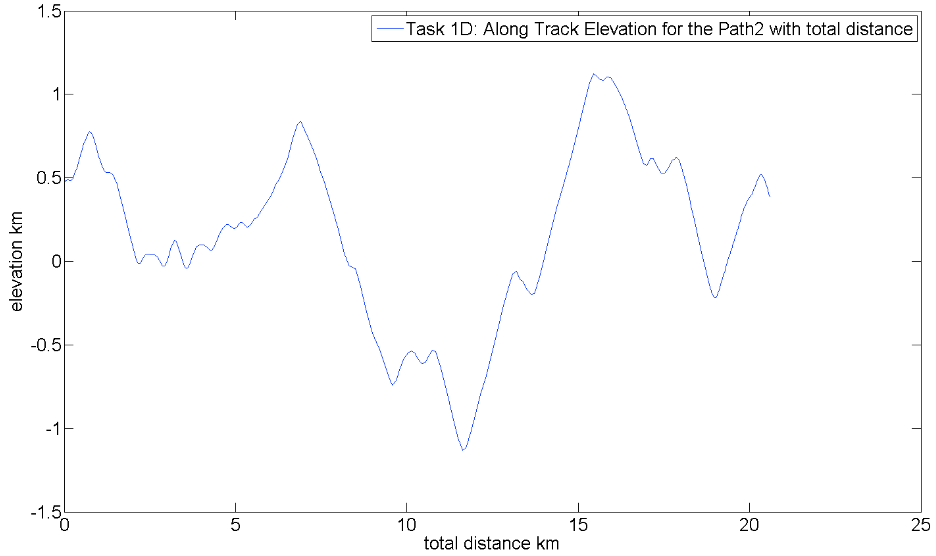

Task 1D: Along Track Elevation for the Direct Path with total distance
elevation km
total distance km
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
2
4
6
8
10
12
14
16
18
20


Task 1D: Along Track Elevation for the Path1 with horizontal distance
elevation km
horizontal distance km
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
2
4
6
8
10
12
14
16
18
20

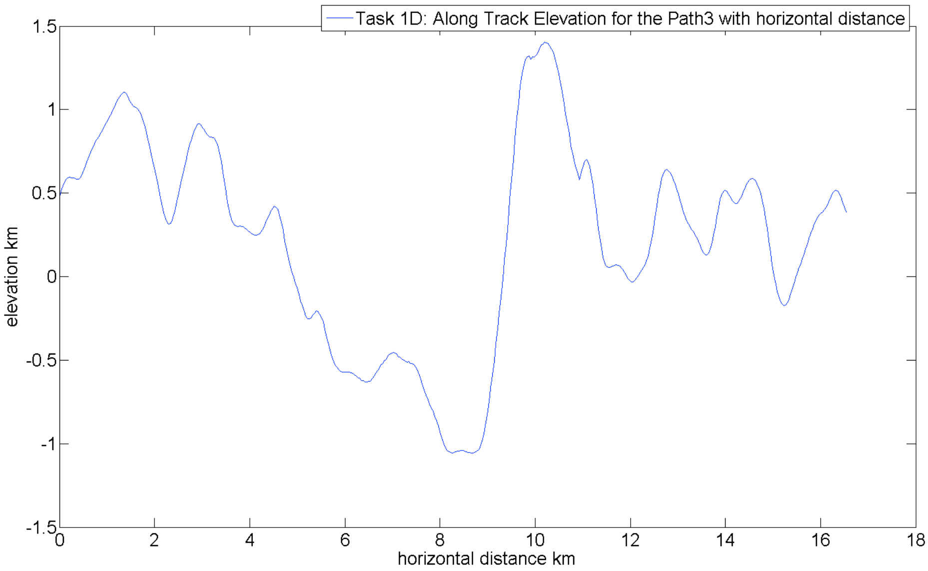

Task 1D: Along Track Elevation for the Path1 with total distance
elevation km
total distance km
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
5
10
15
20
25

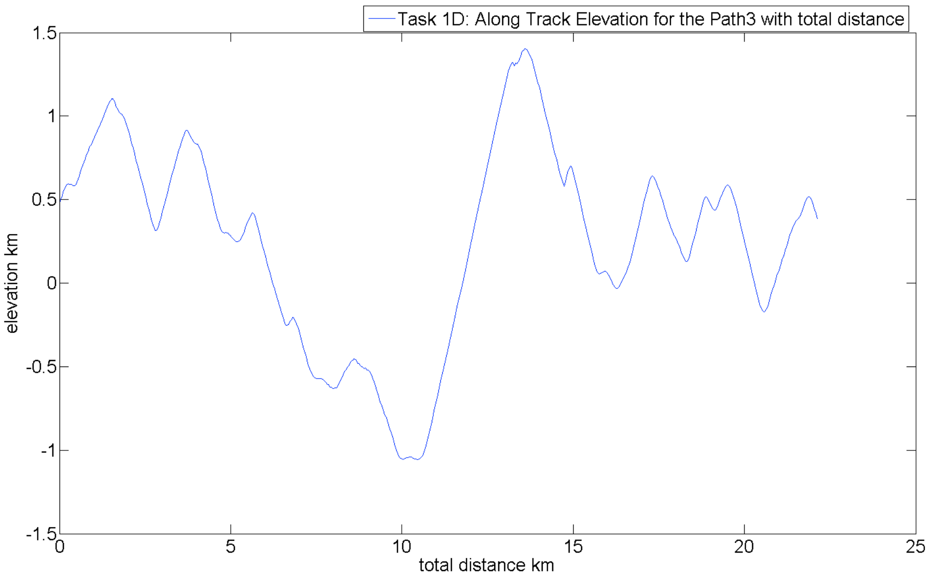


Task 1D: Along Track Elevation for the Path2 with horizontal distance
elevation km
horizontal distance km
1.5
1
0.5
0
-0.5
-1
-1.5
0
2
4
6
8
10
12
14
16
18



Task 1D: Along Track Elevation for the Path2 with total distance
elevation km
total distance km
1.5
1
0.5
0
-0.5
-1
-1.5
0
5
10
15
20
25





These eight figures show the distance versus elevation. The difference between the graph with toal distance and the graph with horizontal distance is that total distance is usually bigger than horizontal distance and besides the trend for each path between total and horizontal distance is almost the same. From the graph, we can see that path2 has more peaks along the path and the direct path seems to be more flat than others.

2.Re-compute the paths with 200%, 50% and 25% and 10% of the points per segment and discuss the difference. You many find tabulating the results a useful departure point for the discussion.

200%

Route	Time(s)	Distance(km)	plusElev.	minusElev.	max(h)	min(h)

Direct	1.94E+03	19.4265059	6.49546956	6.581128394	1.264760758	-2.043324102
Paht1	2.17E+03	21.74776564	4.908154605	4.993813438	1.154931347	-0.503664897
Paht2	2.07E+03	19.16072202	5.106776962	5.192435795	1.126133347	-1.134239221
Path3	2.23E+03	22.2745280475674	6.239111513	6.324770346	1.401311662	-1.055690632

50%

Route	Time(s)	Distance(km)	plusElev.	minusElev.	max(h)	min(h)
Direct	1.93E+03	19.28747579	6.417741574	6.503400407	1.259298711	-2.044172752
Paht1	2.14E+03	21.39785091	4.502445379	4.588104213	1.147092843	-0.461127896
Paht2	2.04E+03	19.16072202	4.978099087	5.06375792	1.118607718	-1.133179126
Path3	2.21E+03	22.0800839095652	6.147750485	6.233409318	1.403079489	-1.055582664

25%

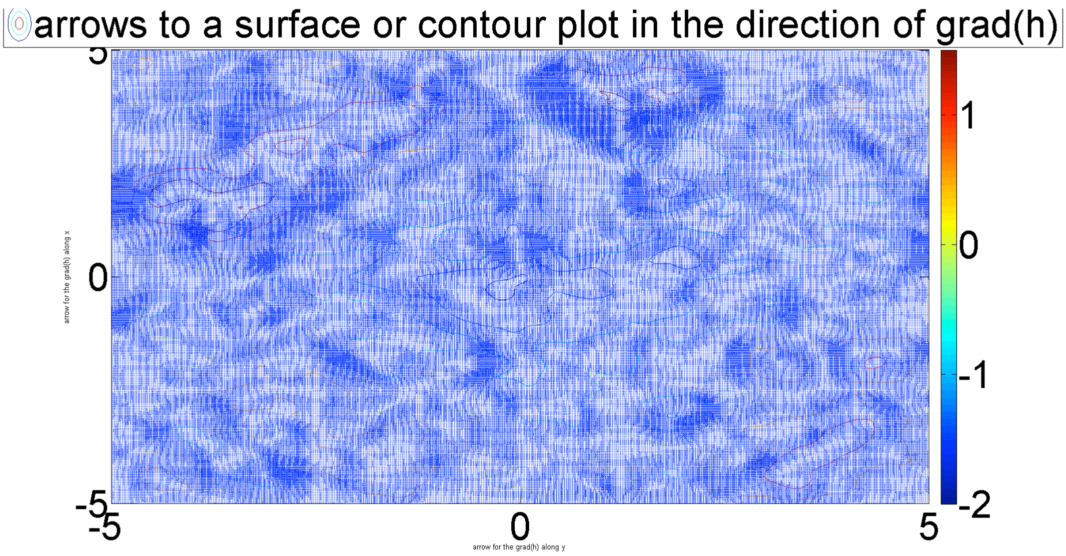
oute	Time(s)	Distance(km)	plusElev.	minusElev.	max(h)	min(h)
irect	1.90E+03	19.01908133	6.2518646	6.337523433	1.261583656	-2.043955473
aht1	2.06E+03	20.63300516	3.976302317	4.061961151	1.151881256	-0.503708607
aht2	2.01E+03	19.16072202	4.749801218	4.835460052	1.098007766	-1.106456959
ath3	2.19E+03	21.9247152290454	6.034945228	6.120604061	1.402035986	-1.055008711

10%

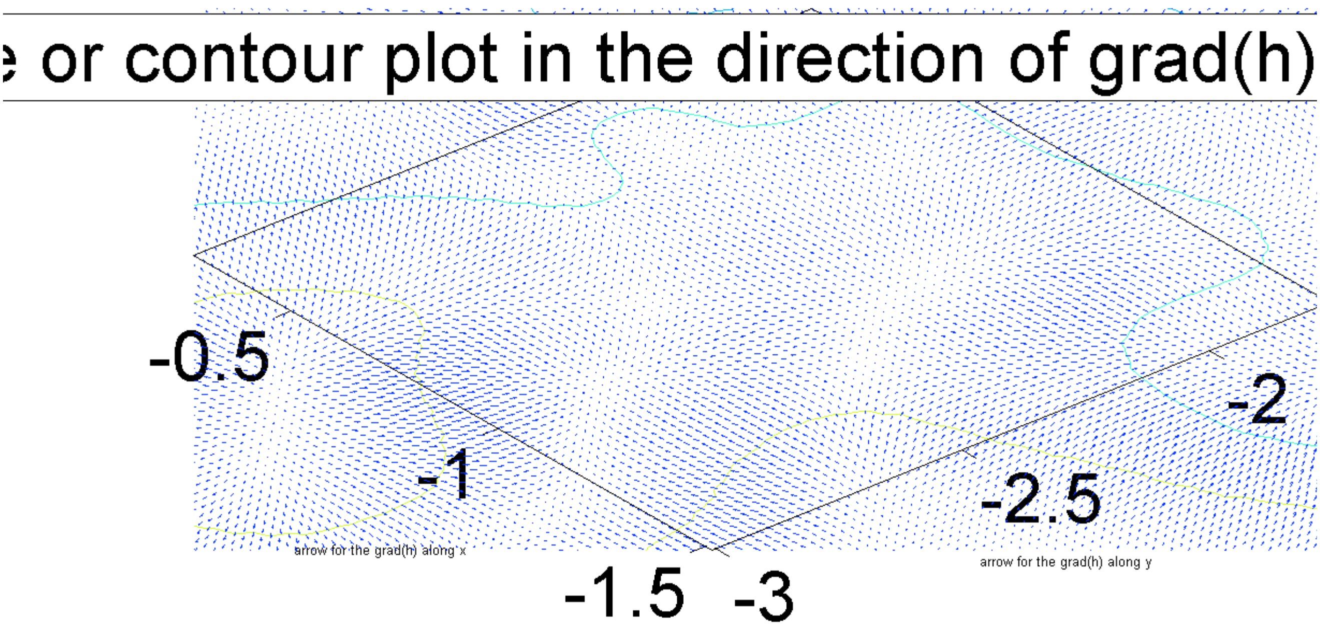
Route	Time(s)	Distance(km)	plusElev.	minusElev.	max(h)	min(h)
Direct	1.78E+03	17.84787666	5.473723422	5.559382255	1.259298711	-1.953718182
Paht1	1.97E+03	19.67570946	2.974675037	3.06033387	1.151881256	-0.340433333
Paht2	1.92E+03	19.16072202	3.852238329	3.937897162	1.124103121	-0.73943993
Path3	2.11E+03	21.0931881371062	5.512815453	5.598474286	1.391082257	-1.053528372

From the above table, We can see that with more points, each path takes more time, longer distance,less +elevation and -elevation. However, both max and min h follow the trend that it becomes bigger first and then it become smaller.

3.Add arrows to a surface or contour plot in the direction of -grad(h) with appropriate magnitude along the path. Discuss what significance these arrows have with respect to an ideal path.



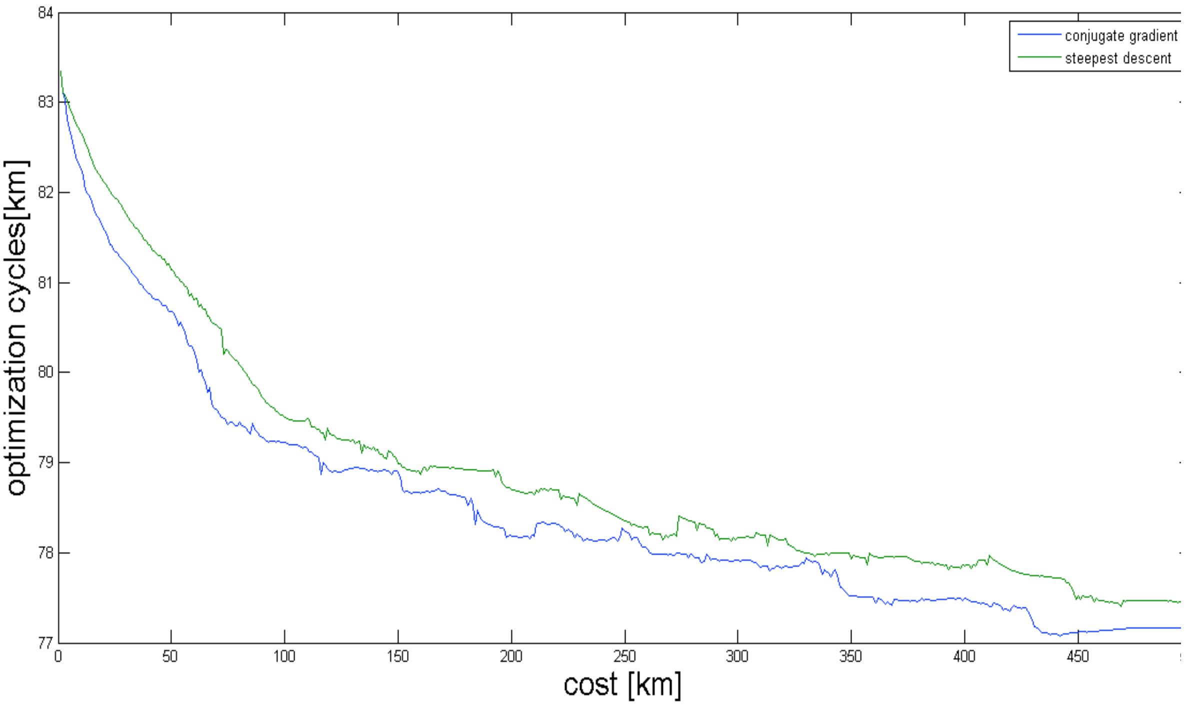
These arrows indicate the direction of the change of elevation along the horizontal distance.



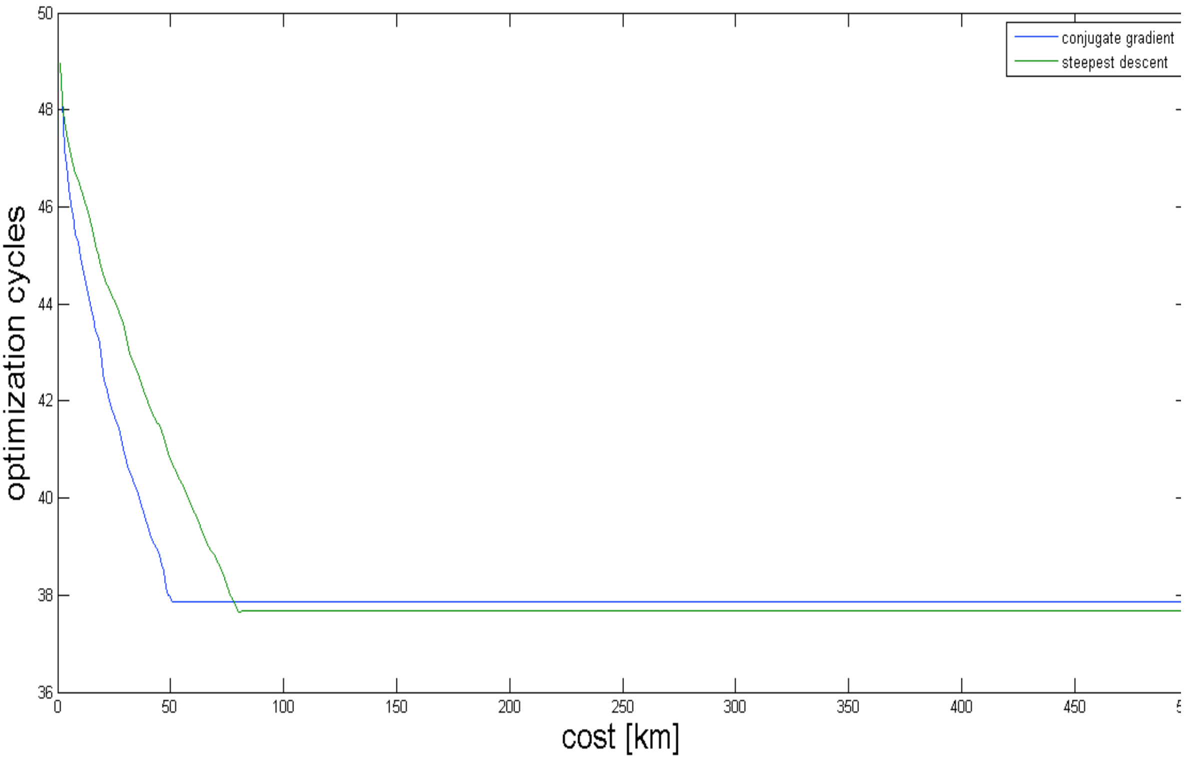
This is the zoomed in version so that you can see the arrows.

Task 2

1. Include and succinctly discuss the tables and figures for parts A-D of task
2. Make sure all the figures are have axis labels and legends where appropriate.



This figure is Cost v. Optimization cycles for the Direct Path.For the direct path,the conjugate gradient and steepest descent does not converge within 500 cycles.



This is Cost v. Optimization cycles for the Path1.For the path1, the conjugate gradient and steepest descent converges at around 80 cycles.

1. Make a table of the cost before and after the optimization and the number of cycles needed for the optimized to converge. The table should include all the same quantities as the table in Task 1C for both the original and optimal paths, as well as the min and max elevations on the path for both the original and optimized paths.

Route	Time(s)	Distance(km)	plusElev.	minusElev.	max(h)	min(h)
Direct	1.93E+06	17.84787666	5.473723422	5.559382255	1.263315724	-2.042469512
Paht1	1.97E+06	19.67570946	2.974675037	3.06033387	1.151881256	-0.499168831
Paht2	1.92E+06	19.16072202	3.852238329	3.937897162	1.124103121	-1.130899753
Path3	2.11E+06	21.09318814	5.512815453	5.598474286	1.401906785	-1.055579823
Direct_opt	1.88E+03	18.83762911	5.411817973	5.497476806	1.226172204	-2.011619304
Paht1_opt	2.02E+03	20.18039032	2.592727074	2.678385908	0.471390065	-0.239225834
Paht2_opt	1.98E+03	19.76292818	3.735353881	3.821012715	1.110282567	-0.9705264
Path3_opt	2.08E+03	20.82995785	4.25763123	4.343290063	1.140757534	-0.897474664

This table shows that for direct path and path3, optimized paths take less time ,distance,+elevation,-elevation than non-optimized path. All the optimized path have smaller max and min than the non-optimized one. Although the optimized path1 and path2 take longer time, distance, all the optimized path actually take smaller +elevation and -elevation, which means they consume less energy and expend less money.

Route	cost
Direct	3.04E+02
Paht1	1.48E+02
Paht2	2.04E+02
Path3	3.46E+02
Direct_opt	77.16008049
Paht1_opt	37.86562883
Paht2_opt	51.73211448
Path3_opt	59.91508154

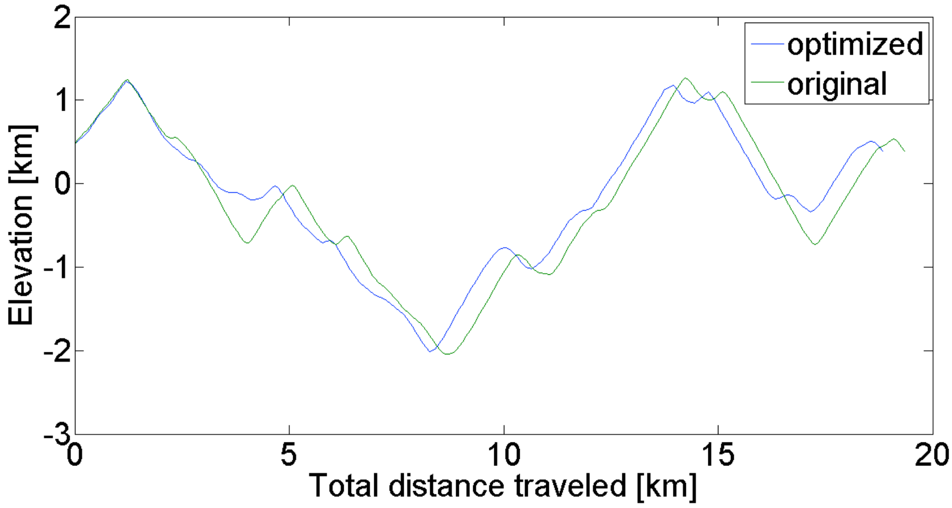
This table shows the cost of paths before and after optimized. From the table, we can see that optimized path saves money.

Route	converge cycle
Direct_st	630
Paht1_st	80
Paht2_st	84
Path3_st	1230

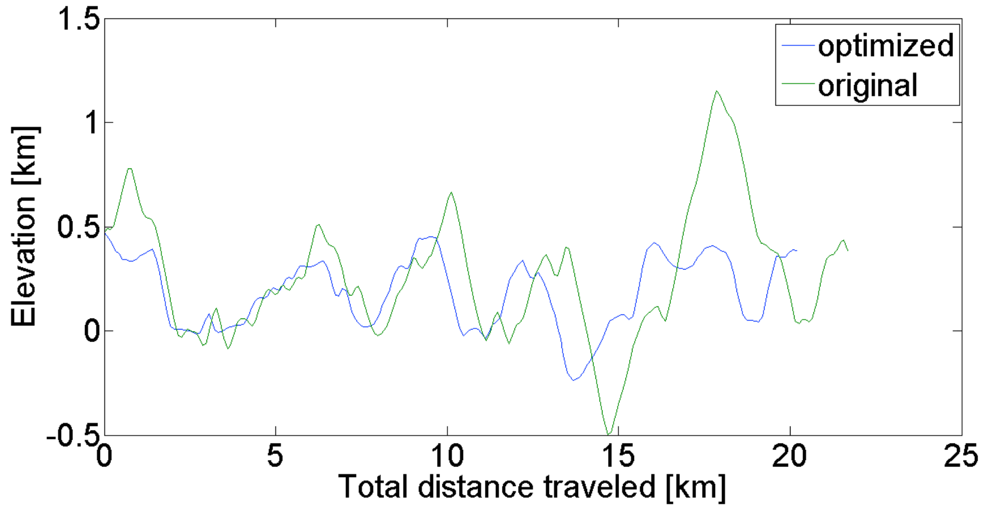
Direct_cg	465
Paht1_cg	50
Paht2_cg	50
Path3_cg	850

This table show the number of convergence cycle and for both steepest descent and conjugate gradient path. The steepest descent optimization takes more cycles to converge to an optimized path compared to the conjugate gradient, as evidenced by comparing the data, and the blue line and the green line. However, both converge to about the same value.

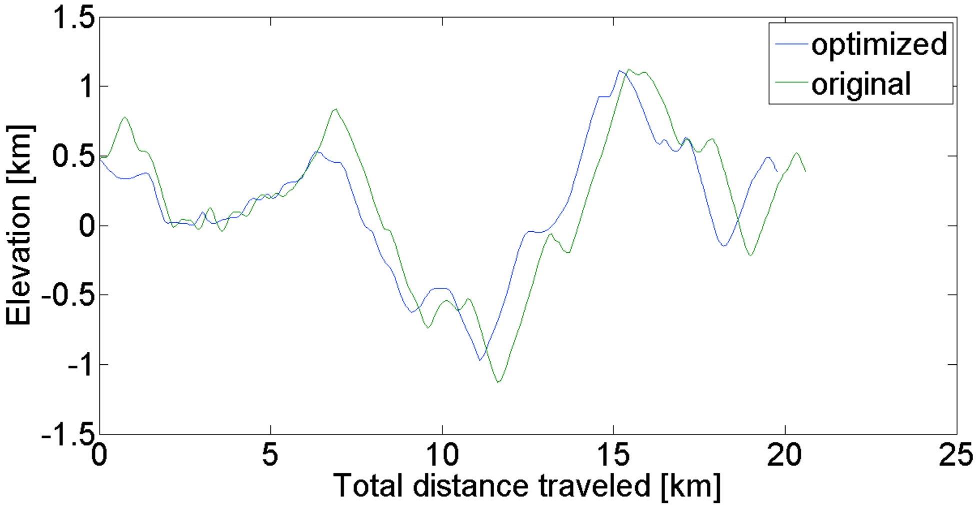
2.Plot the elevation versus along-track distance for each path both original and optimized. (x-y) plots. Each Path should be in a separate plot



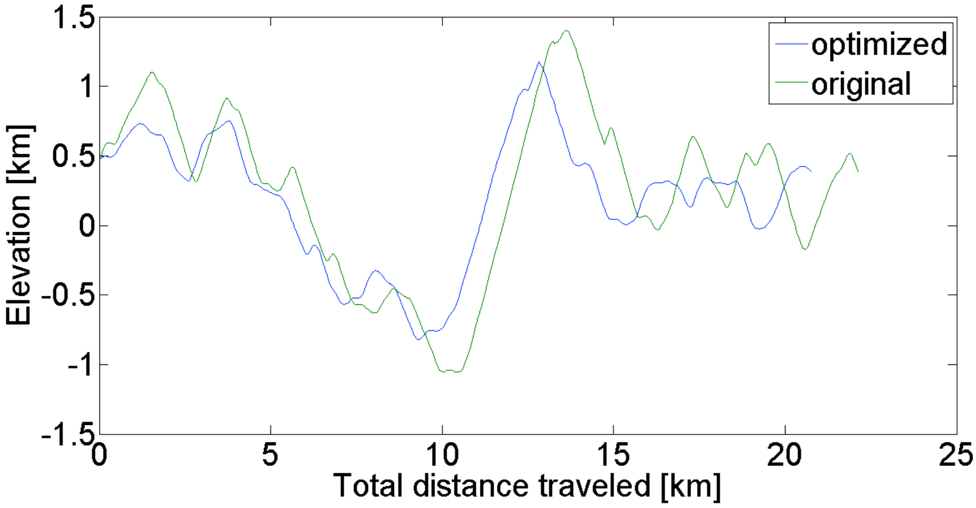
elevation versus along-track distance for direct path



elevation versus along-track distance for path1



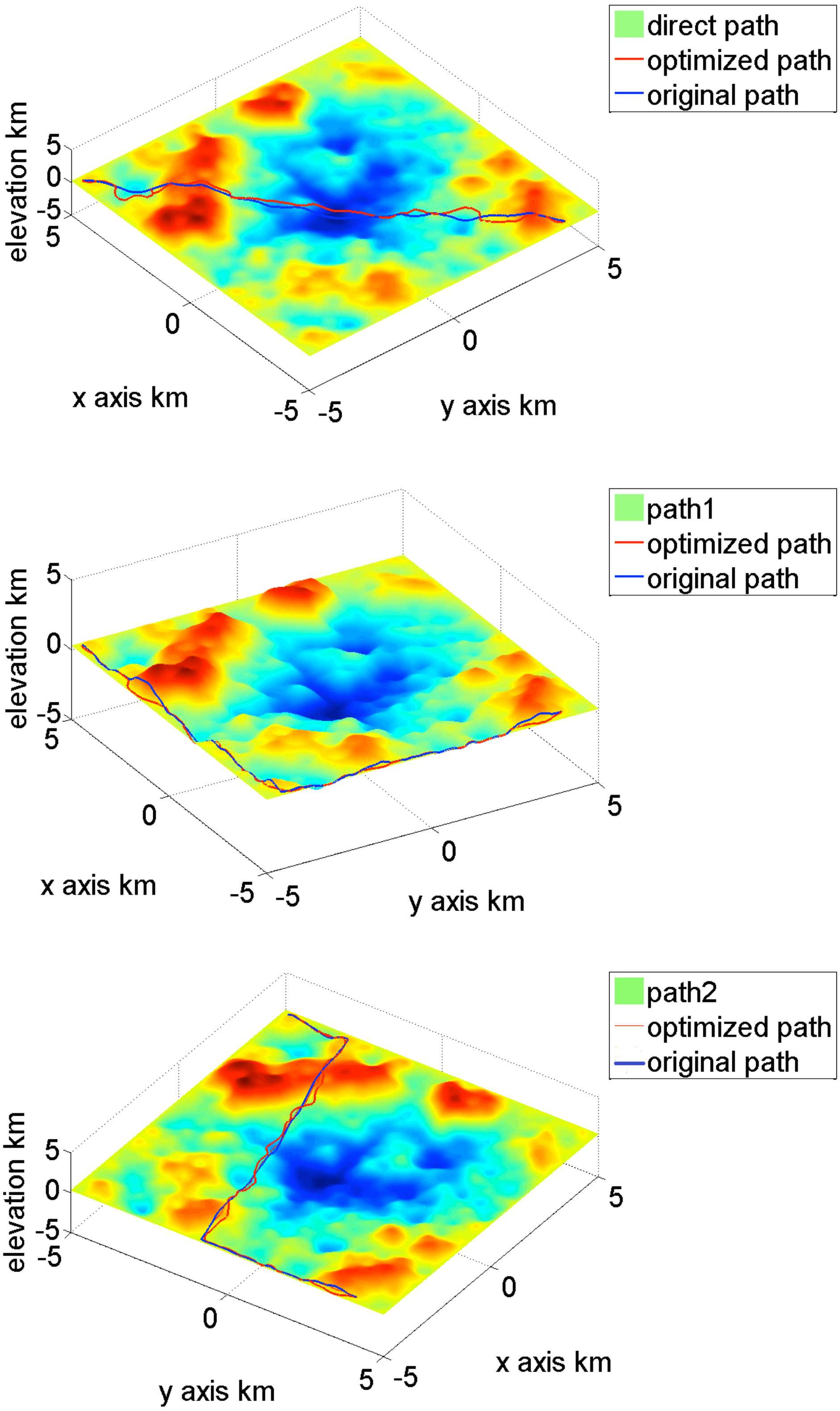
elevation versus along-track distance for path2

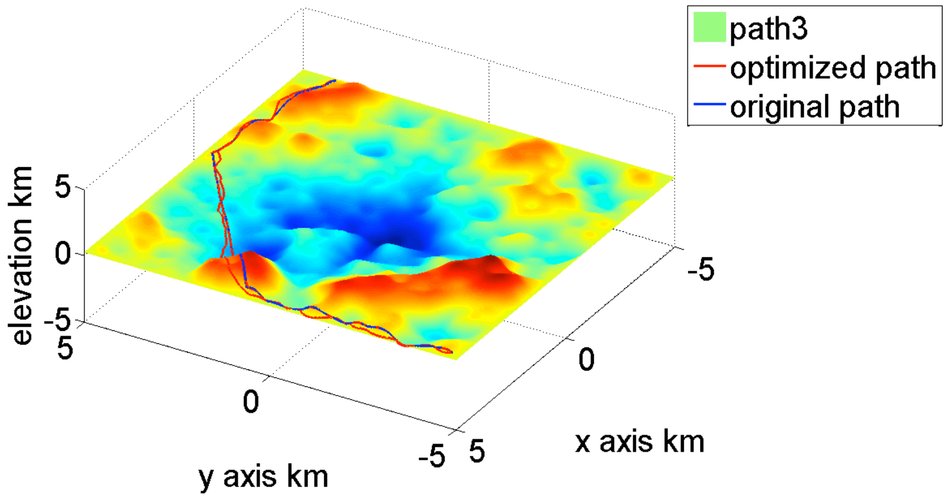


elevation versus along-track distance for path3

The above three figure show that original paths have longer distance than the optimized paths. Moreover, the original and the optimized path diverge more seriously if the distance is bigger. The optimized path seems to be less fluctuated on the elevation than the original one.

3. Plot both the original and optimal paths on a three-dimensional surface plot.


direct path
optimized path
original path
elevation km
5
0
-5
5
0
x axis km
-5
-5
0
5
y axis km
path1
optimized path
original path
elevation km
5
0
-5
5
0
x axis km
-5
-5
0
5
y axis km
path2
optimized path
original path
elevation km
5
0
-5
-5
0
y axis km
5
-5
0
5
x axis km



2. Re-compute the optimized paths with 50% and 25% and 10% of the points per segment and discuss the differences observed. In particular, the cost savings per iteration, and the changes in the cost J should be compared. Again, you many find tabulating the results a useful departure point for the discussion.

ost[km]	50%	change 50%-25%	25%	change 25%-10%	10%
irect_st	76.2952946	-0.5355	76.83080414	1.8951	74.9356865
aht1_st	38.31309307	0.6046	37.70846583	-0.0559	37.7643717
aht2_st	52.33373605	0.2721	52.06166981	-0.4339	52.4955965
ath3_st	57.41725324	1.3497	56.06752484	-0.5473	56.6148563

From the table, we can see that for path1,2,3, the cost first decrease from 50% points to 25% points and then increases from 25%points to 10%points. However, for the direct path,the cost first increases from 50% points to 25% points and then decreases from 25%points to 10%points.

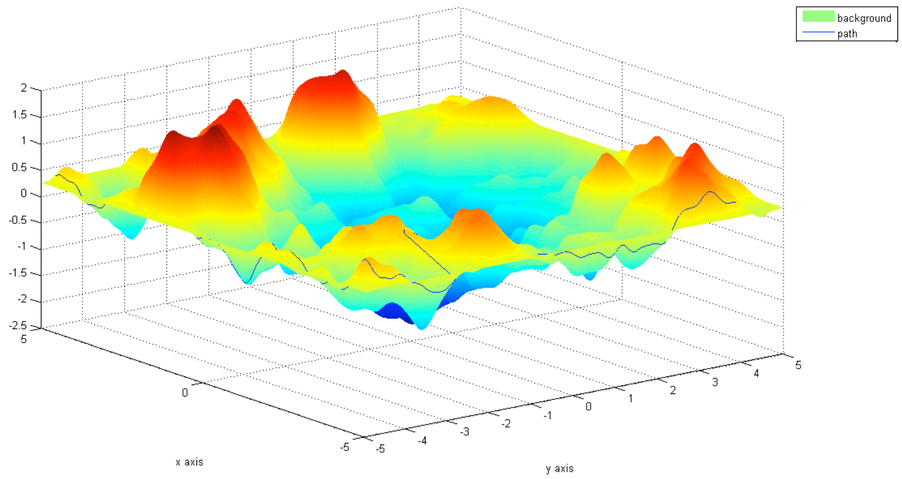
3. Modify descent.m so that the parameter dmax is variable. Then develop an algorithm that such that the path continually refines from 50 points initially to around 250 points over the span of 100 or so iterations. Compare the cost on the same path with 250 initial points and a constant dmax.

cost with modified dmax	cost without
7.07E+09	7.00E+09

From the table, we see that the path with modified dmax costs more than the one without because the modified one refines more points.

I use :`xp=linspace(xs,xe,50);yp=linspace(ys,ye,50);refine=0.991;`
`dmax_ini=0.2;cycles=100;`

Task 3- Option 1



Appendix:

Description for .m file

getProjectInput.m

%This function should be called getProjectInput and take a single %input, the project file name, project.ini. It should have multiple outputs which are: TerrainFile, StartPoint,
%EndPoint, NumWayPoints, and WayPoints (if NumWayPoints is greater than zero). StartPoint and
%EndPoint should be a 1x2 or 2x1 vector. WayPoints should be a NumWayPoints x 2 or 2 x
%NumWayPoints matrix.

computepath.m

This function generate path without optimization.

compute_cost.m

This function computes cost for the path generated by computepath.m optimizewithcost.m

This function give you all the output from descent.m and the cost for every iteration but it works for all the path.

descentwithcost.m

This function give you all the output from descent.m and the cost for every iteration but it only works for direct path.

descent.m

```
% Computes the optimal path (xp,yp) with height hp for a cost function
% of the form J = sum(distance*(1 + (2*g*C*h/V^2)^2) on a terrain.
% Inputs:  xp and yp are the path. The  gravitational constant (g), the constant
%             (C), and velocity are (V) are needed to evaluate the gradient of the cost function.
%           cg=0 is steepest descent and cg =1 is conjugate gradient. cycles is the number
%           of optimization cycles to perform.
% Outputs: The optimal path (xp,yp) with height (hp) after cycles number of either conjugate
%           gradient or steepest descent optimziation.

 %
 %Parameters
 %
 %astep: vary a from 0 to amax using this interval size
 %amax: maximum displacement of any point during each optimization call
 %dmax: The maximum horizontal distance between points.  If d > dmax a point is added
 %      in between.  Used in check_path_points.
 %disp_cost: if set to true the cost after each optimization cycle is output to the
 %      screen
```

descentop.m

The modified descent to make dmax a variable.

boundary.m

This function achieve the goal that the rover is no longer allowed within 0.15 kilometers from any boundary.

block.m

This function make path to avoid circular region with xc,yc as the center of the circle and r as the radius.

calc_gradients.m

%calc_gradients: Calculates the x and y derivatives
 % of a two-dimensional field f.
 %Inputs: x,y are the one-dimensional grids, and f is the two-dimensional field.
 %Outputs: dfdx and dfdy are the two-dimensional fields of the derivatives.
 %
 %Comments: This function uses the central difference approximation and Neuman BCs.

 %preallocate

compute_cost.m

This function computes the cost for paths.

compute_costop.m

This function computes the cost for optimized paths.

computepath.m

This function compute path with the input from path files.

direct_path.m

This function calculates the direct path.

optimizewithcost.m

This function gives you optimized path and cost for every iteration.