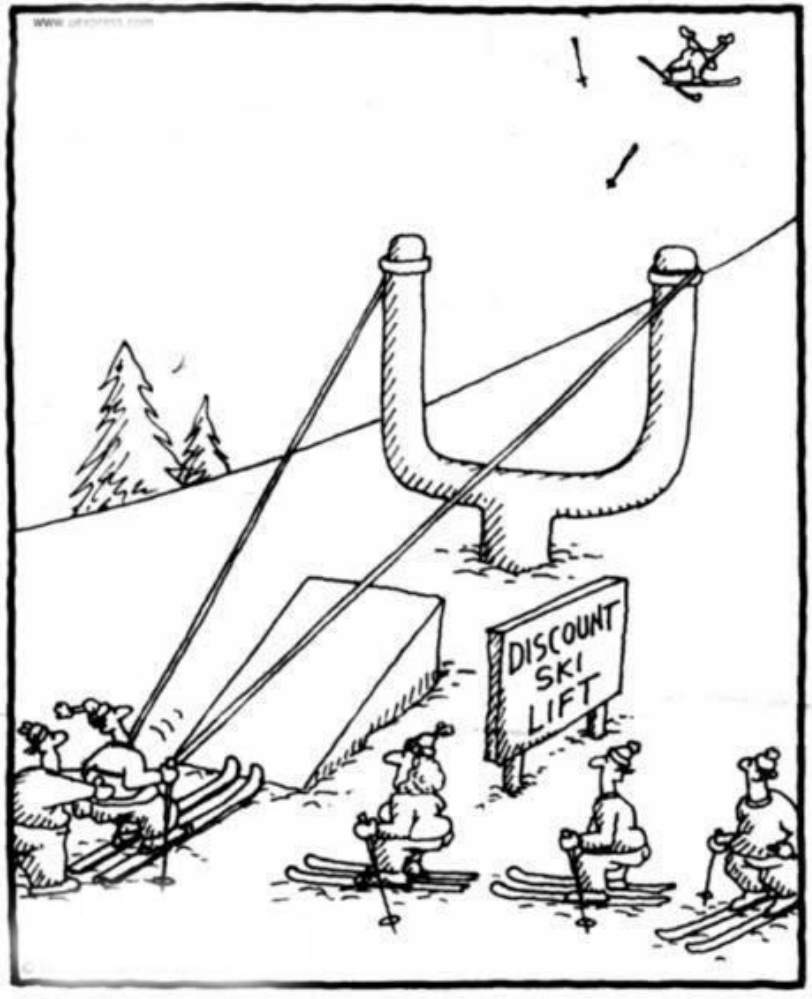
Artificial Intelligence & Small Data

Tjerk Harkema

2017-06-06

Gradient Descent


DISCOUNT
SKI
LIFT

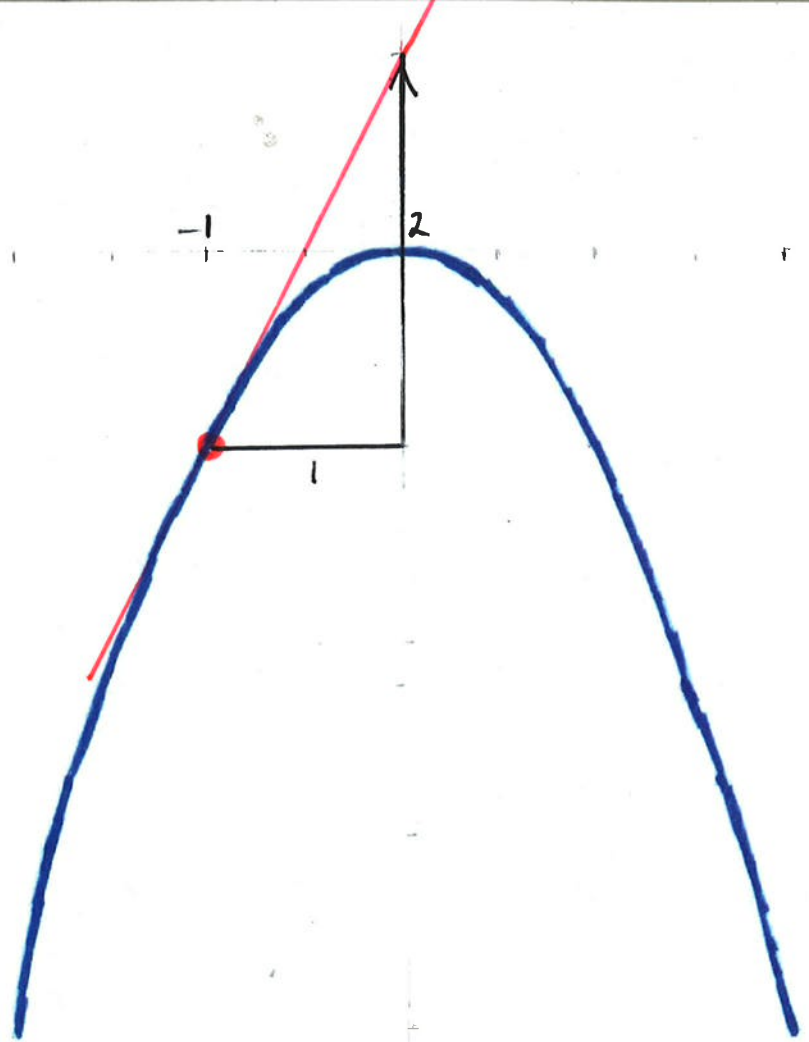

10%

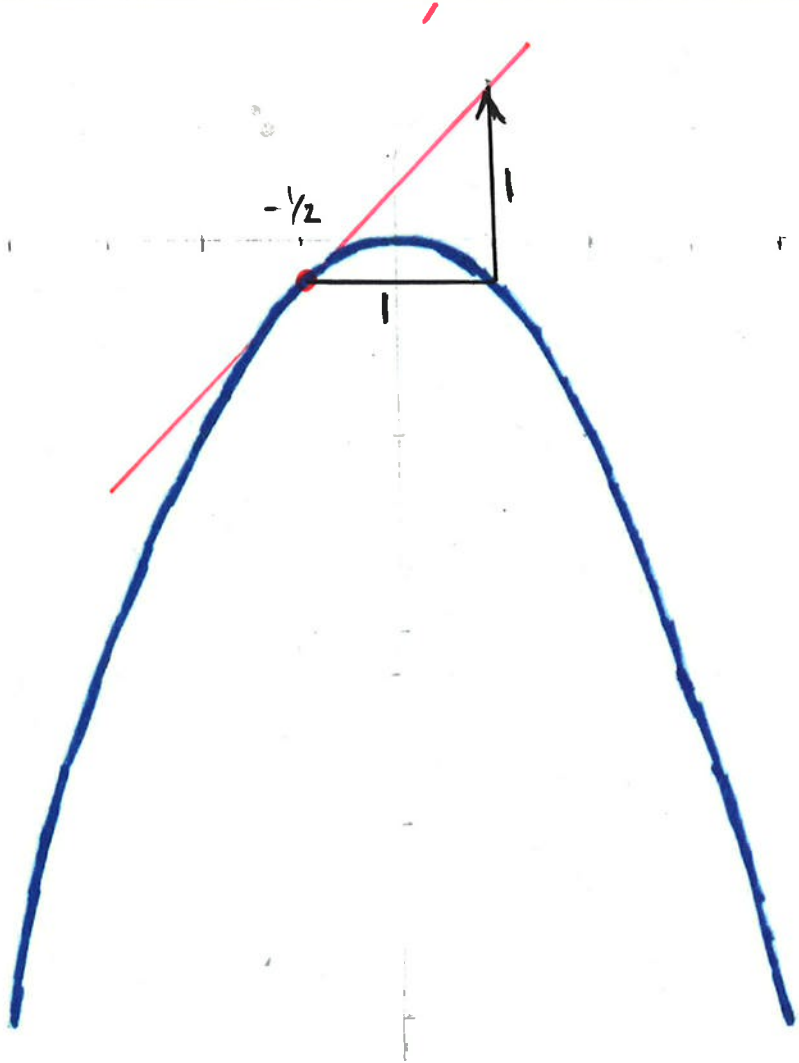
$$\text{slope} = \text{rise}/\text{run}$$

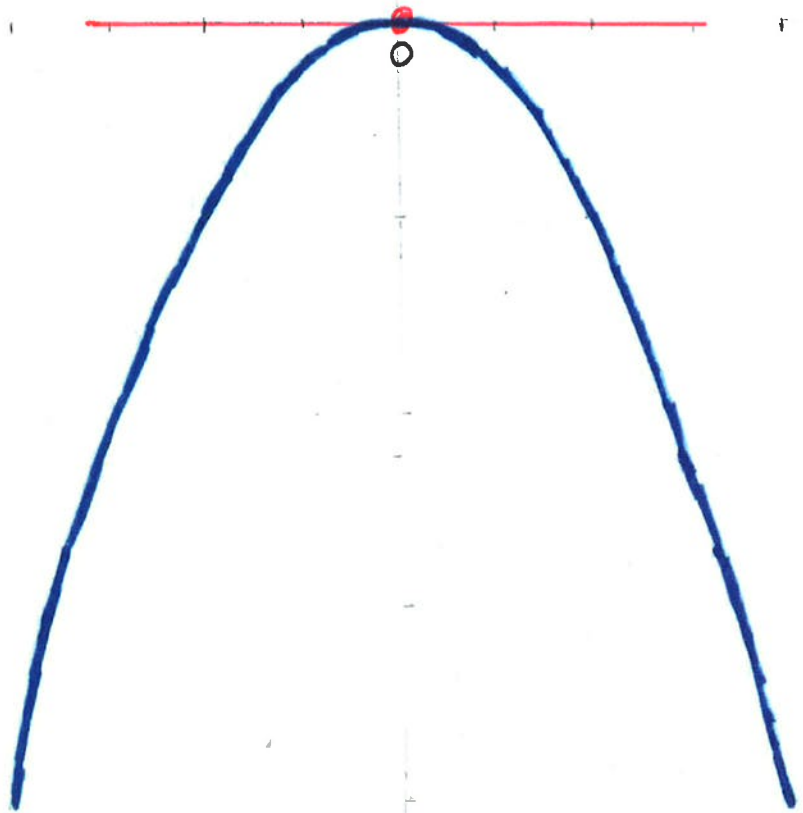

Slope Tree
POSITIVE SLOPE
NEGATIVE SLOPE
ZERO SLOPE
UNDEFINED SLOPE

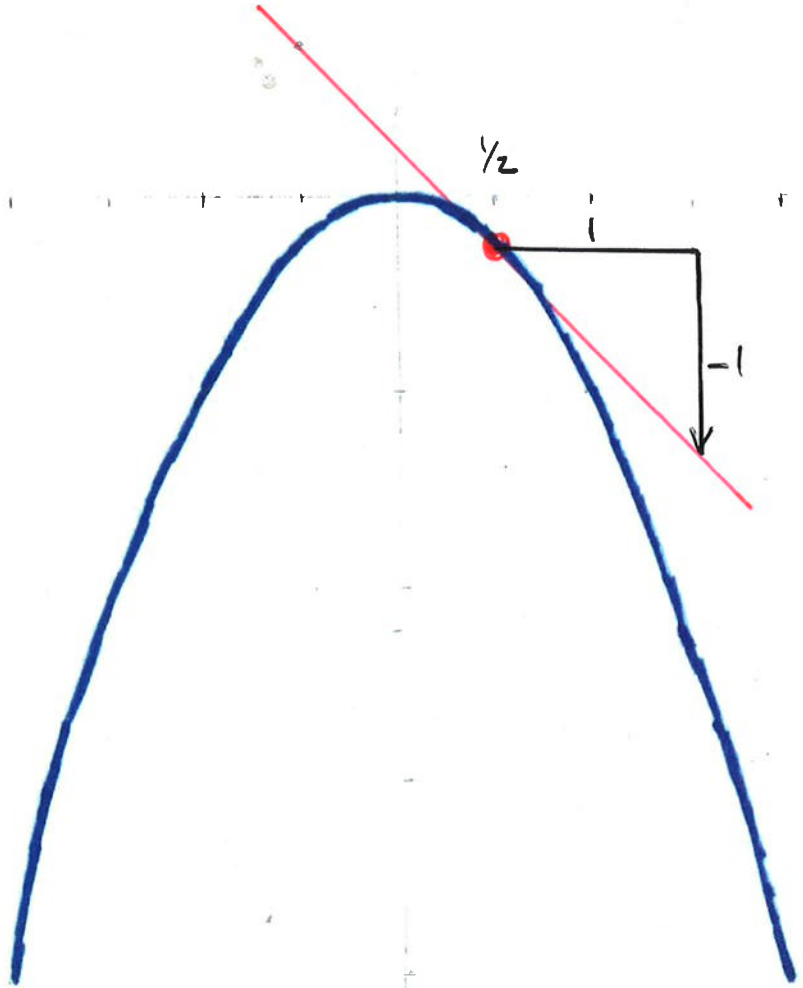
๑๑

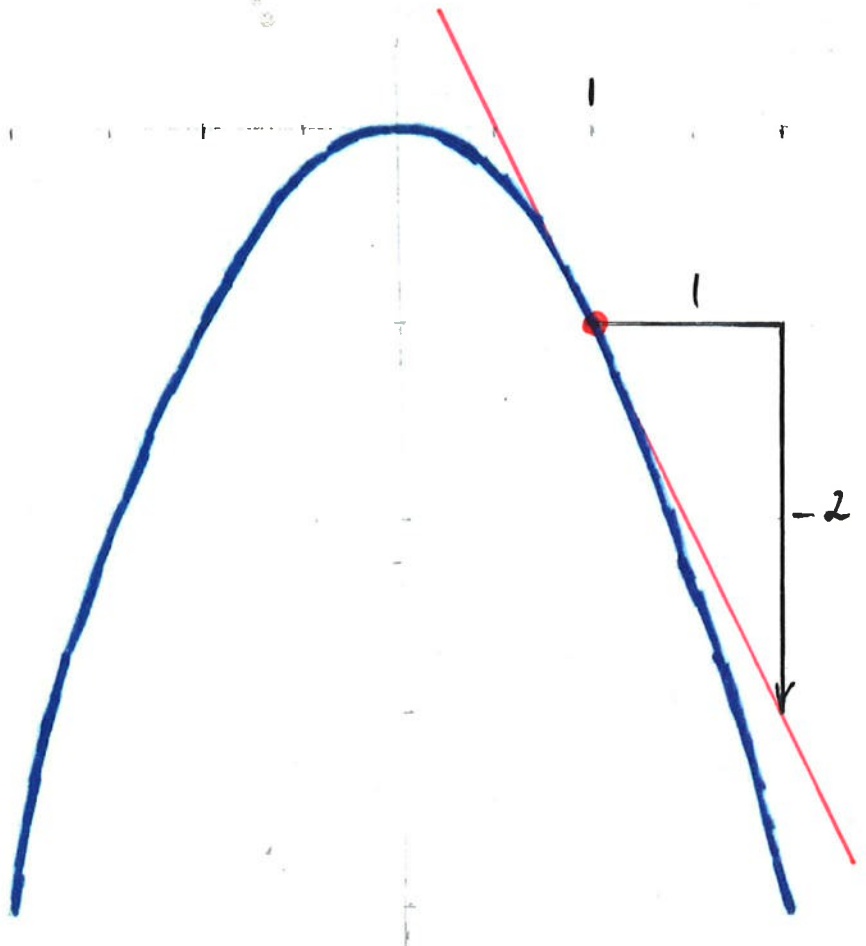

Uphill


-1
2
1

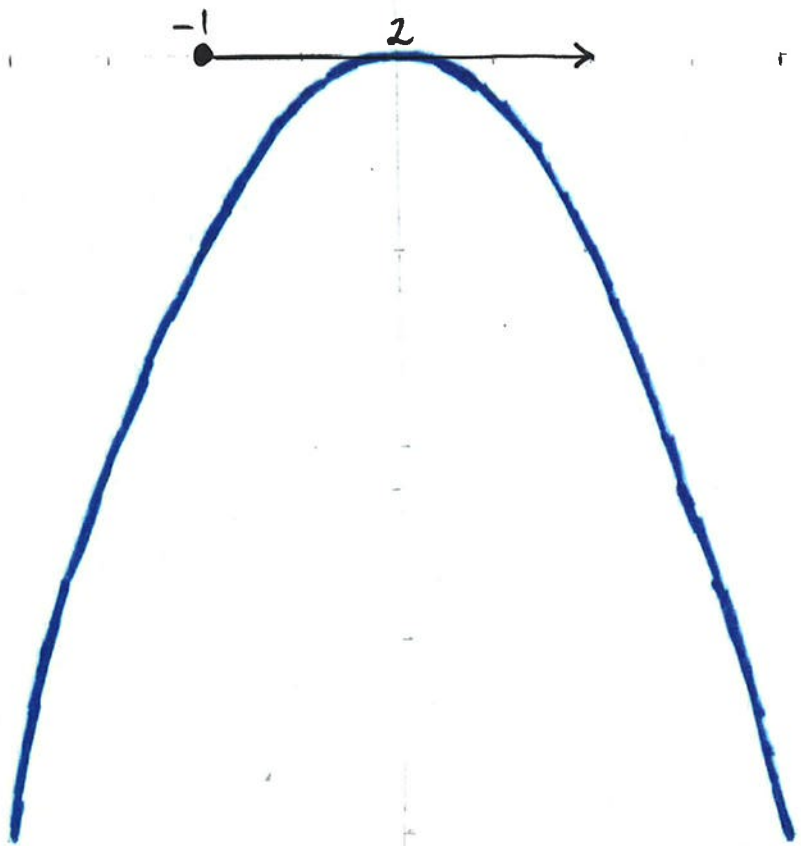

-1/2
1
1

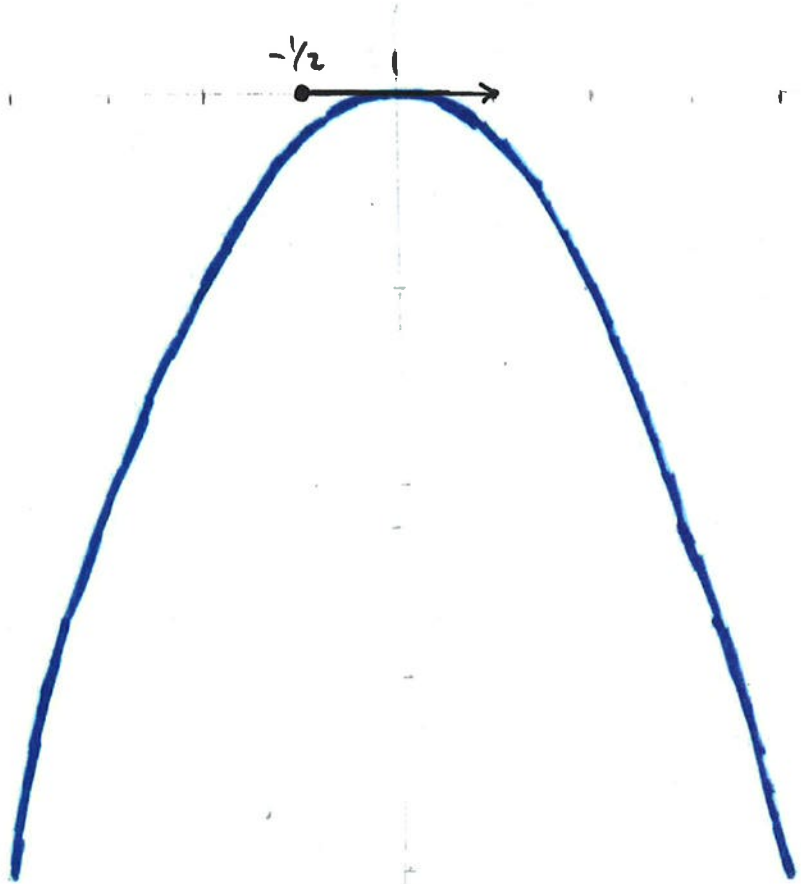

O

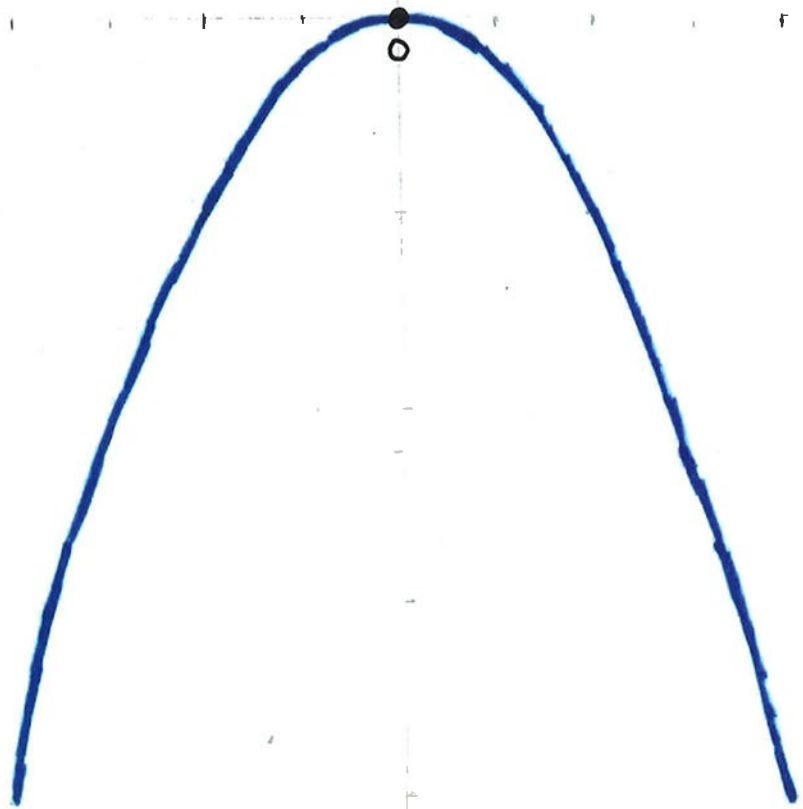

1/2
1
-1

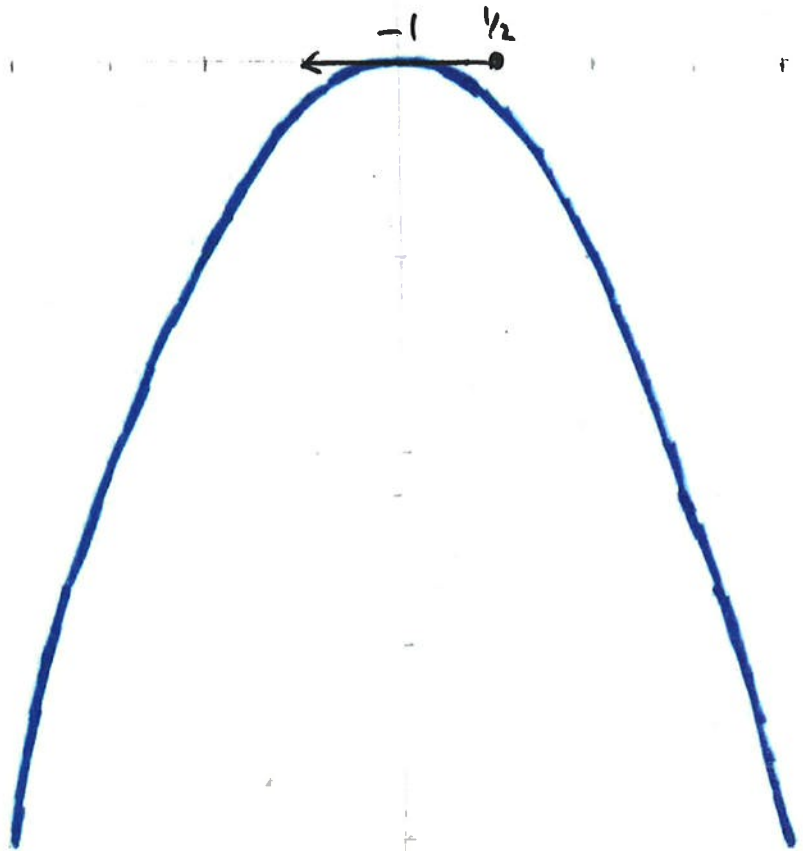

1
1
-2

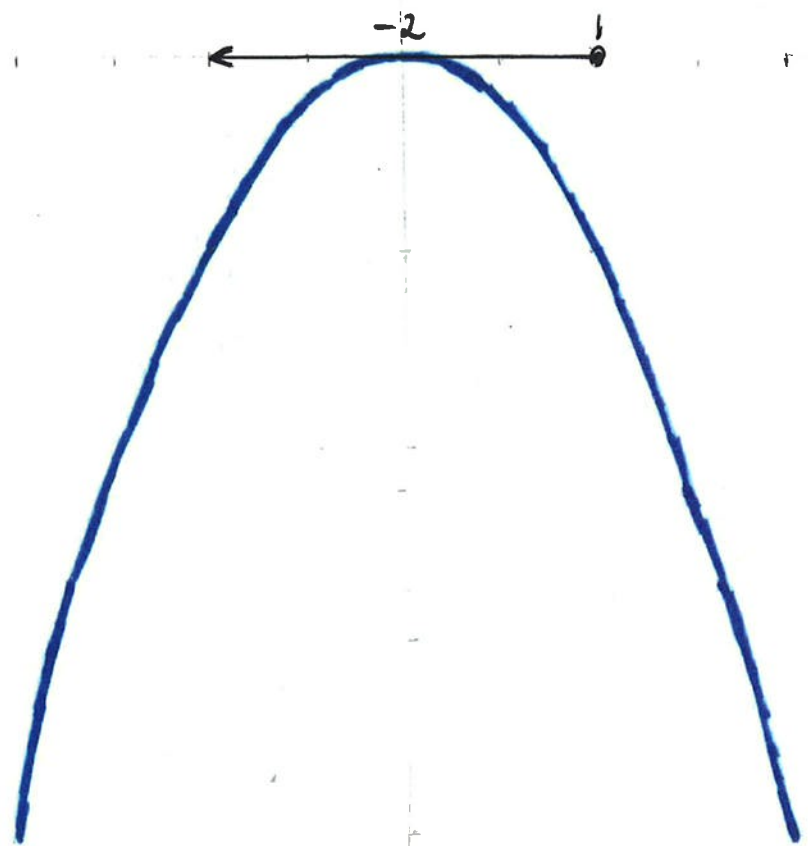
Vector Field


-1
2


-½
1

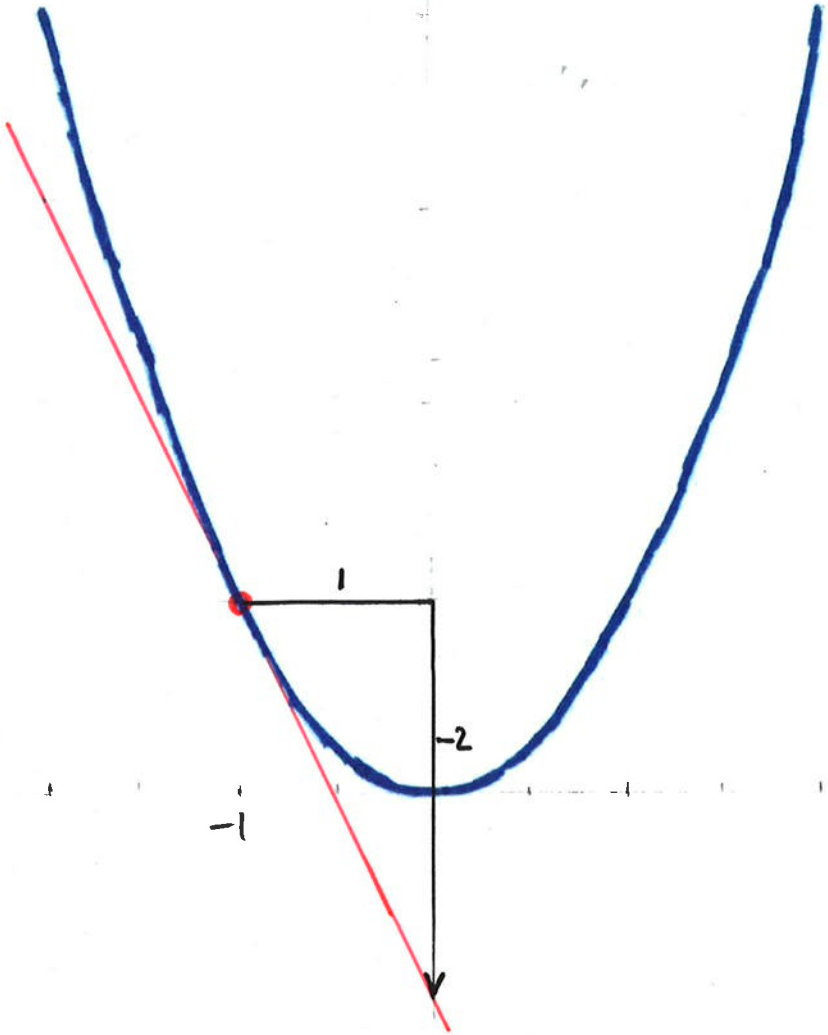

O

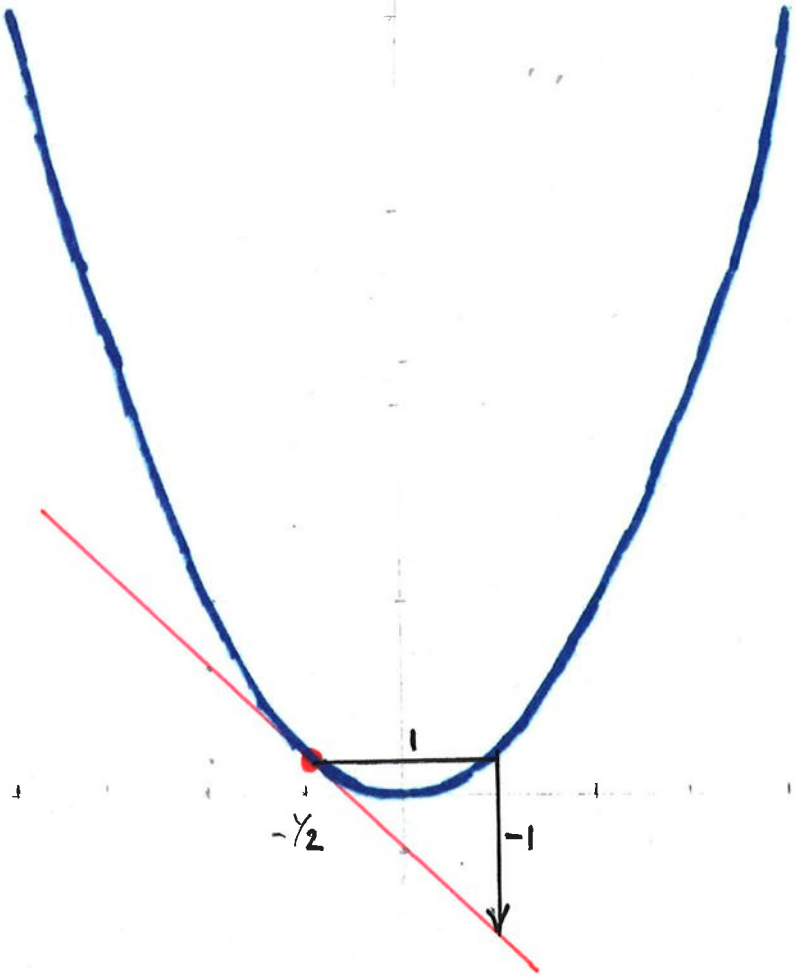

−1
1/2

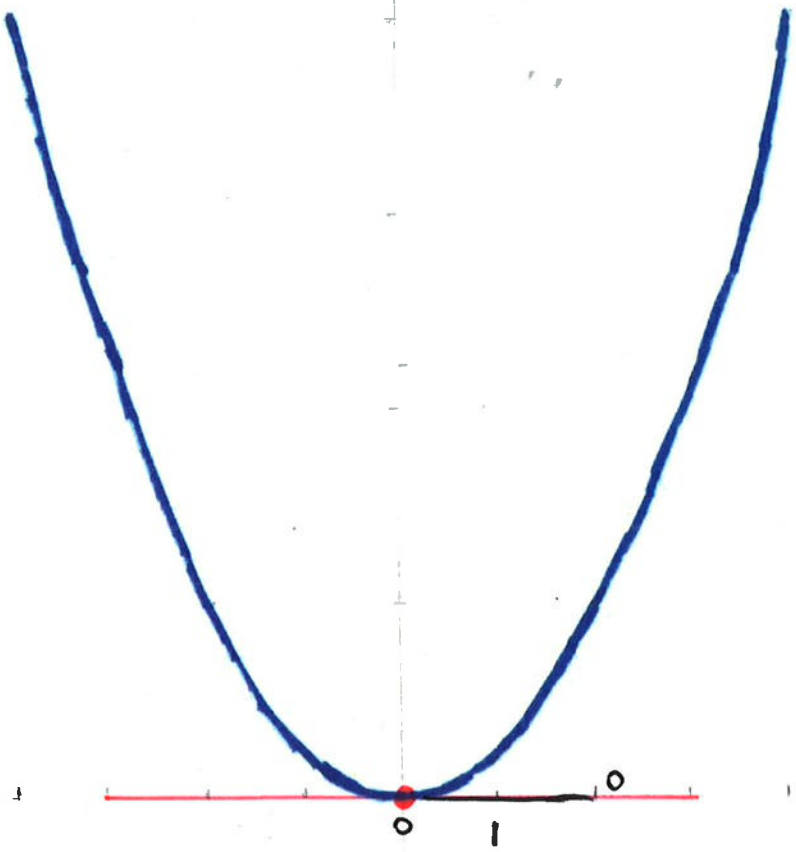

-2
1

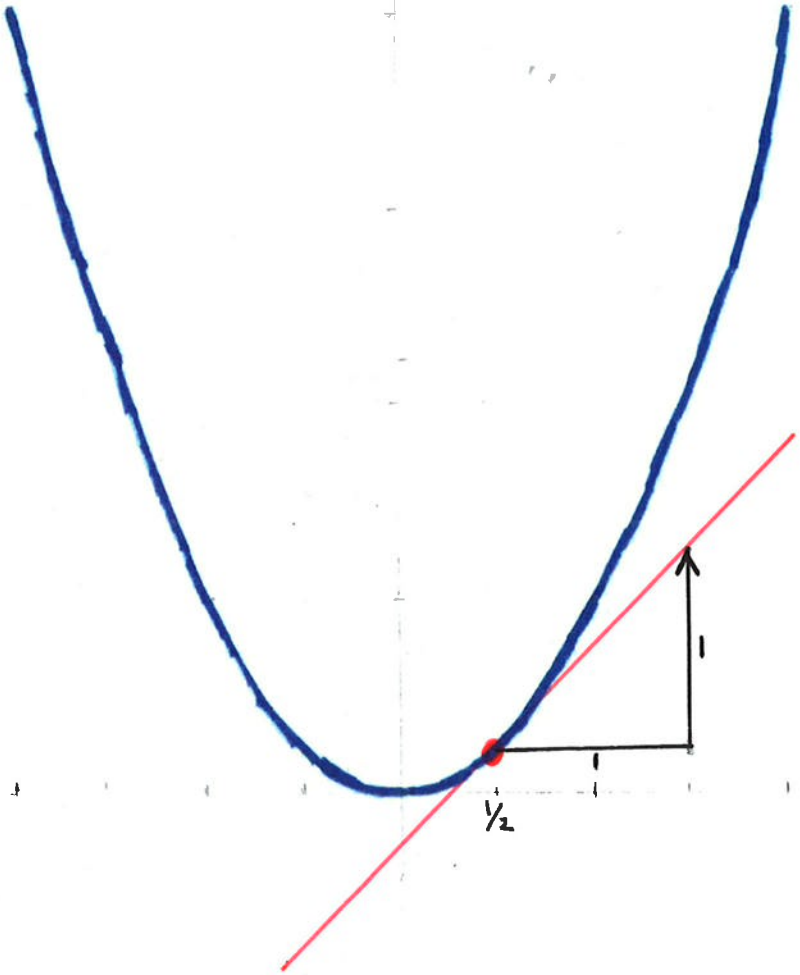

Downhill

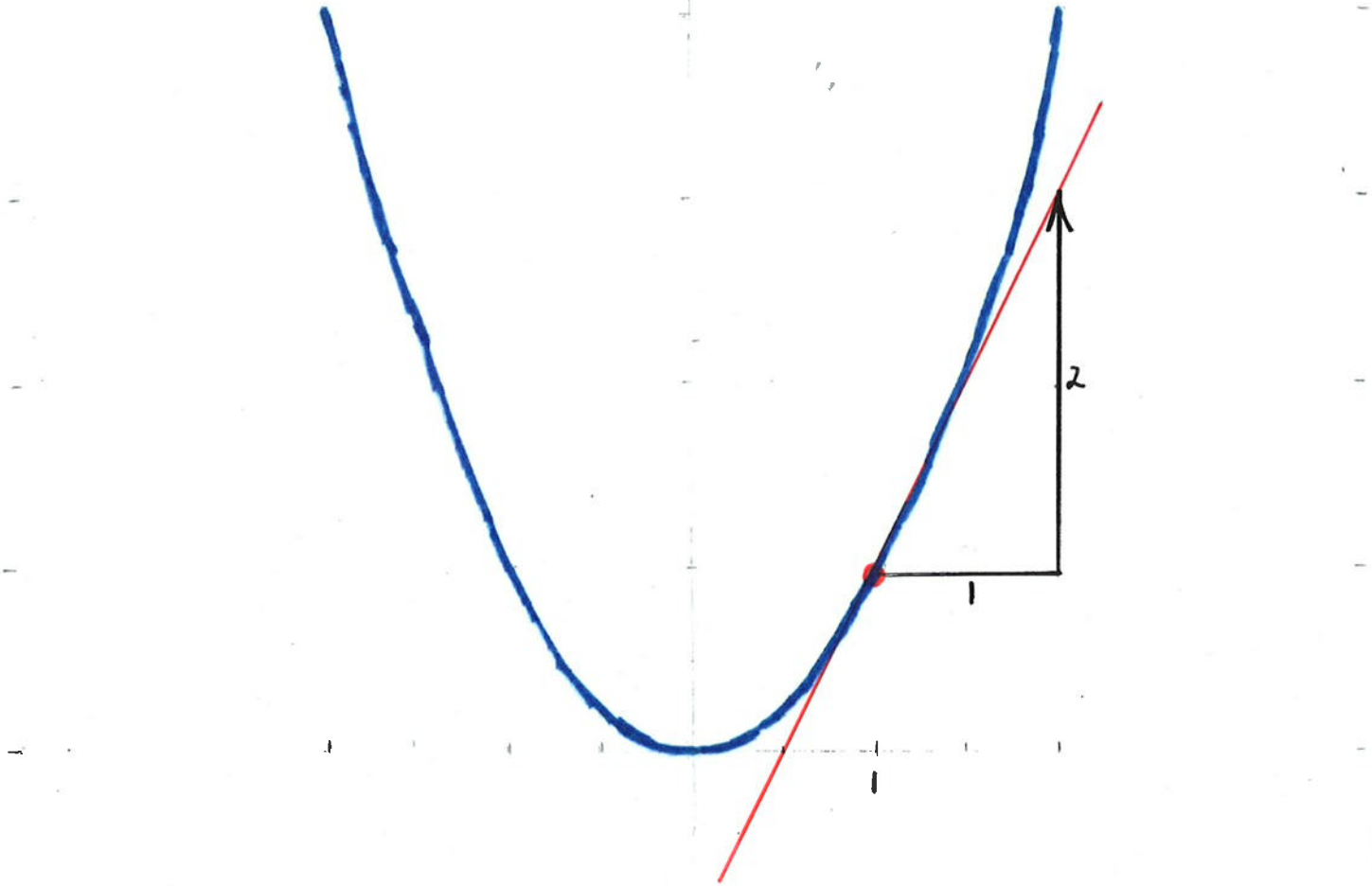

RESTROOM


1
-2
-1

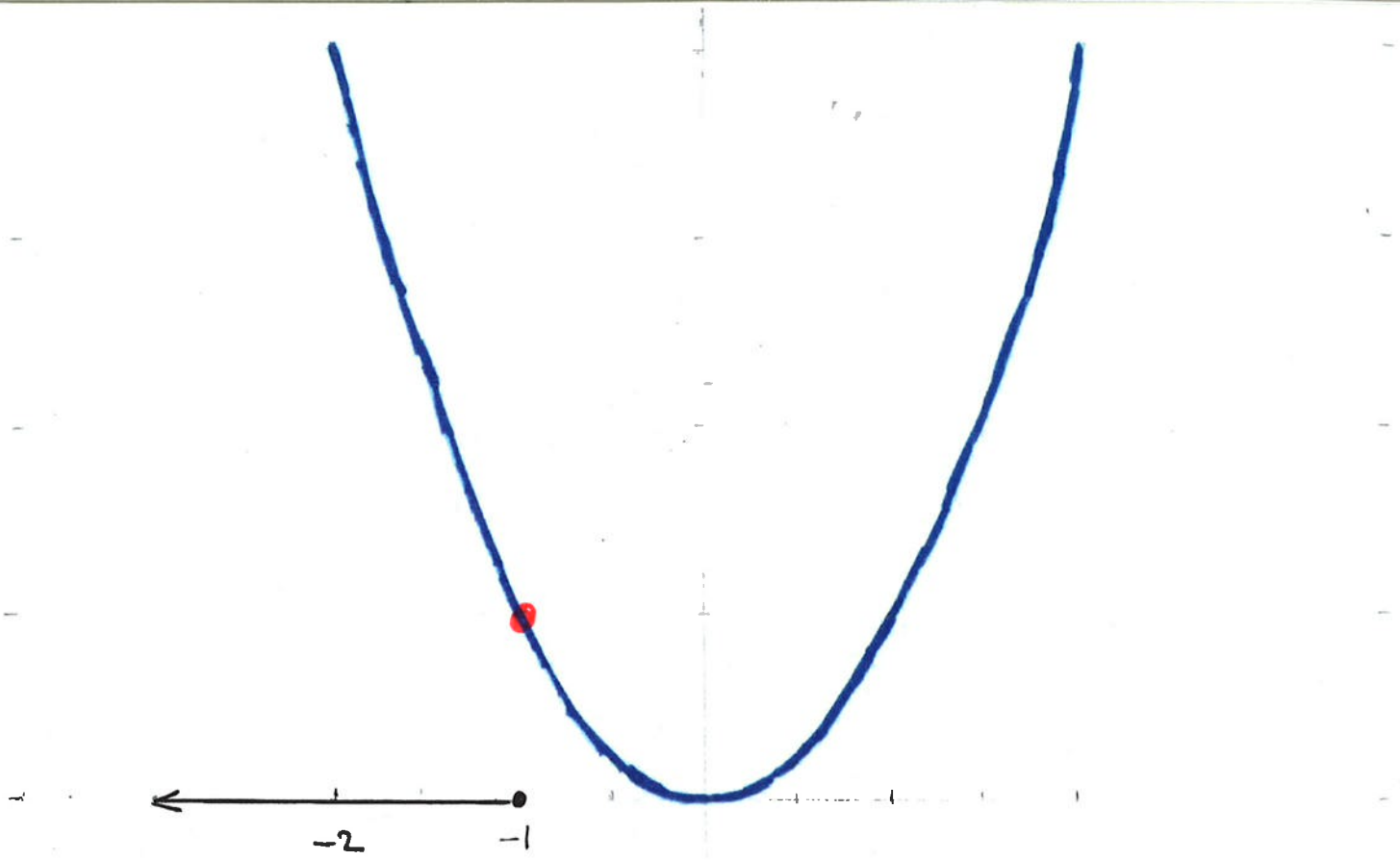

1
-1/2
-1

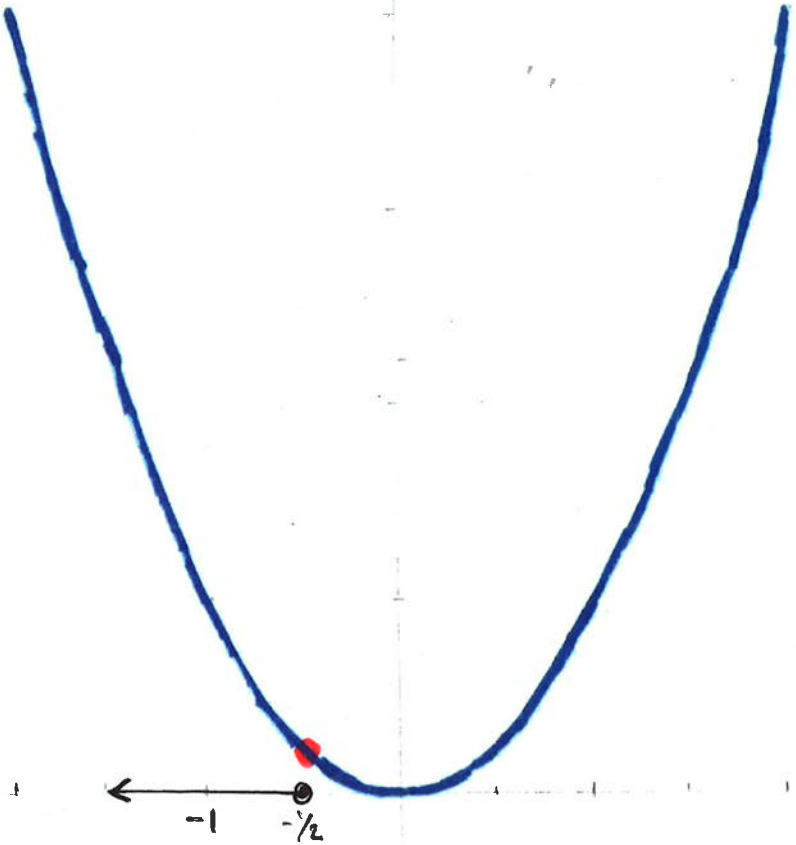

0
1
0

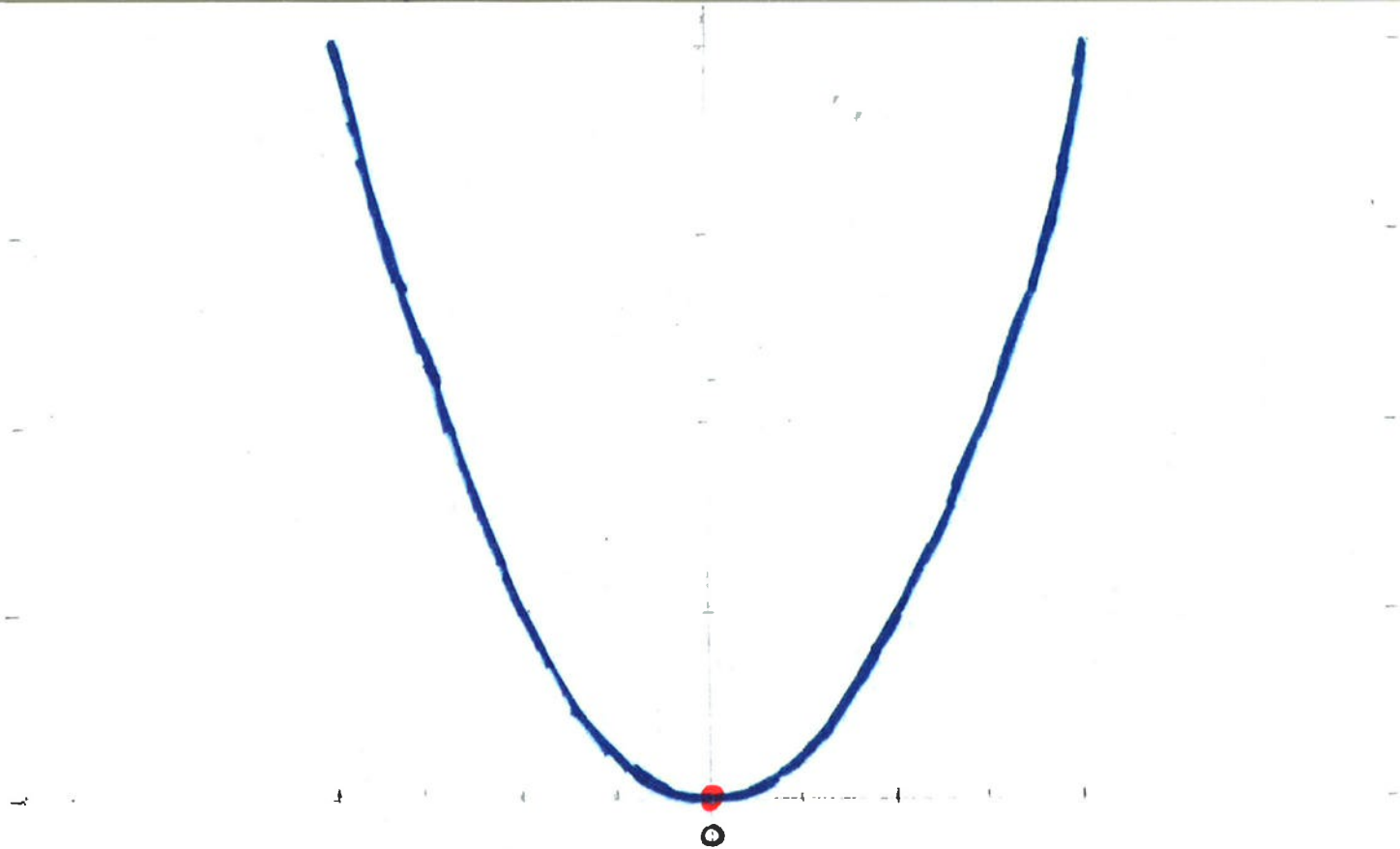

1
1
1/2

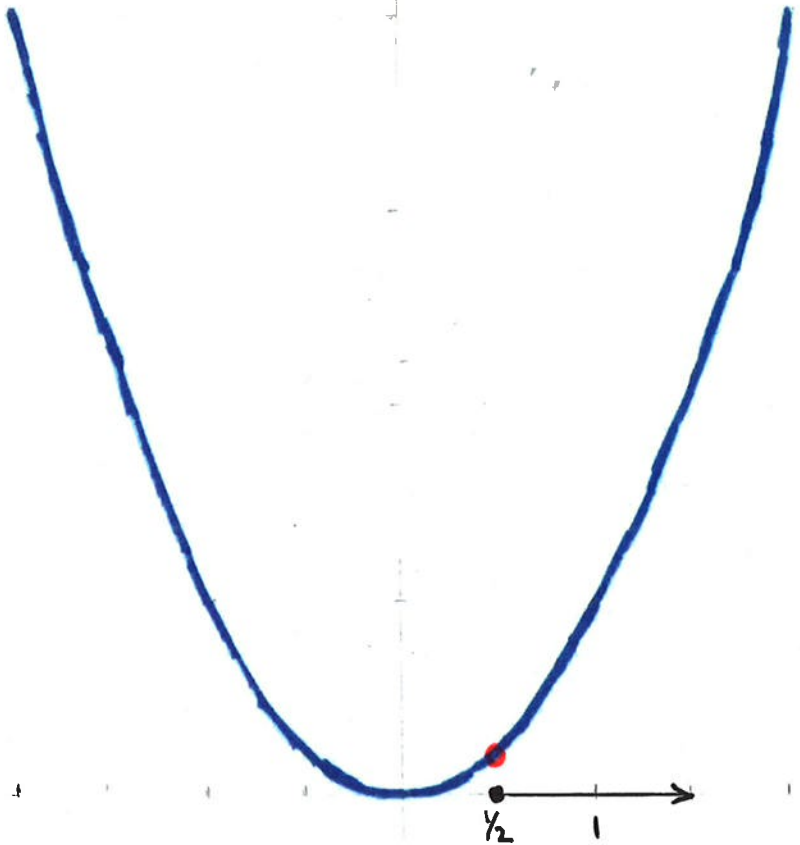

2
1
1

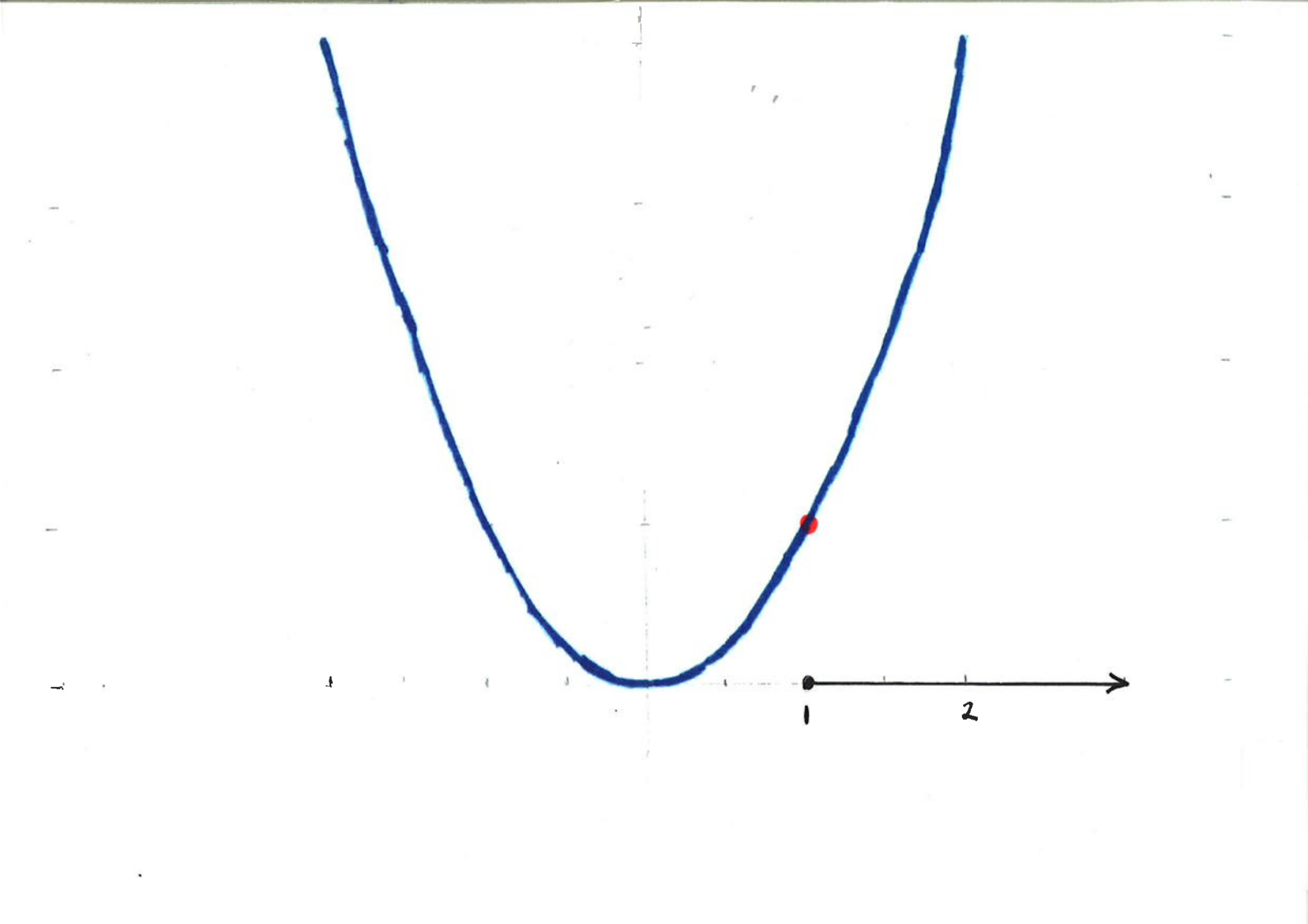
Vector Field


−2
−1


-1
-½

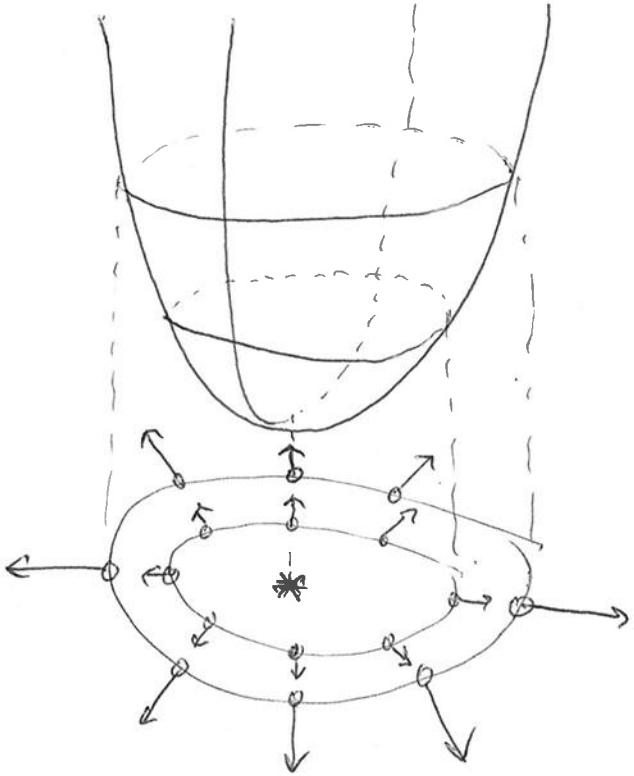

O

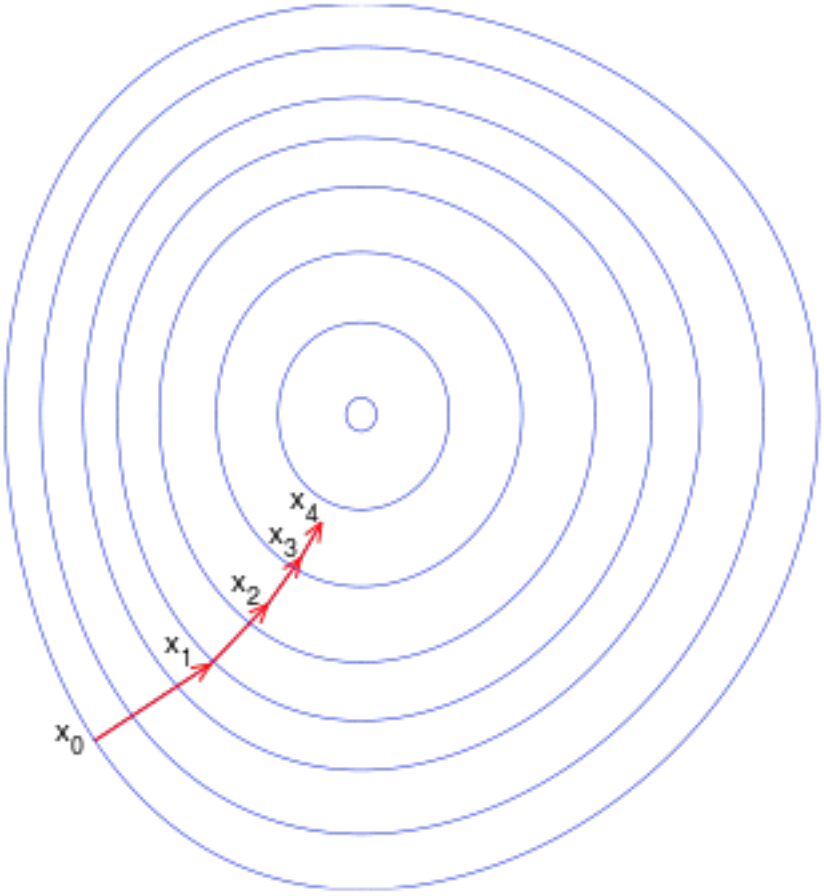

1/2
1

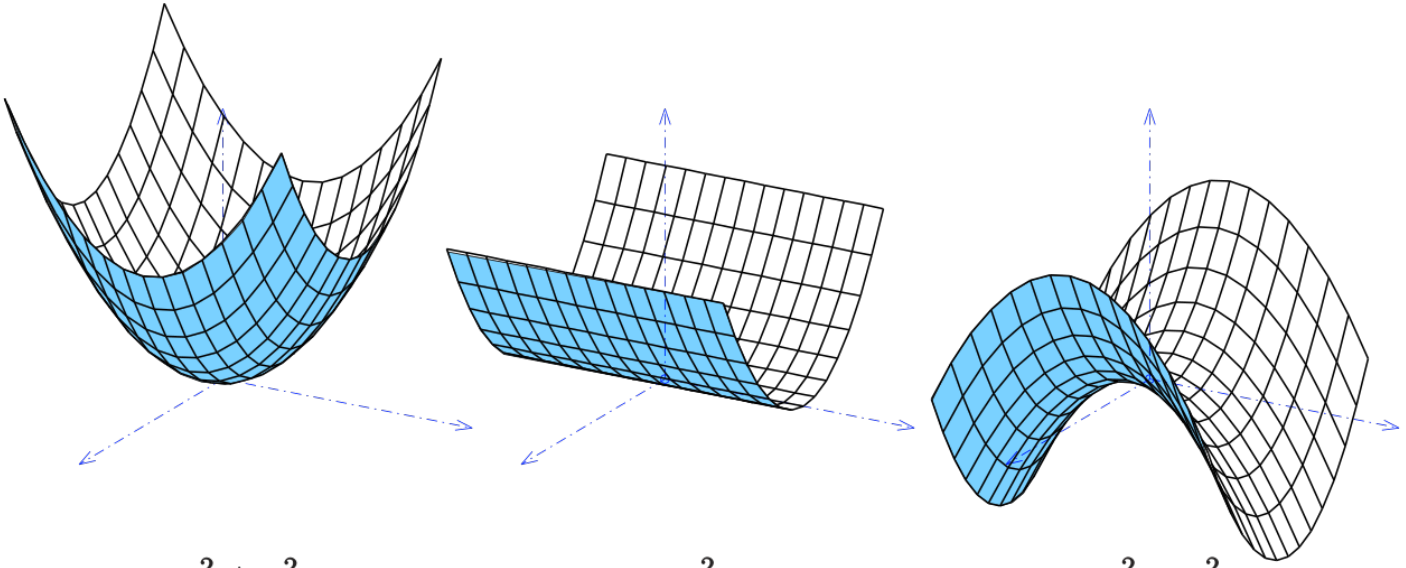

1
2

point → point - fraction*vector


2D

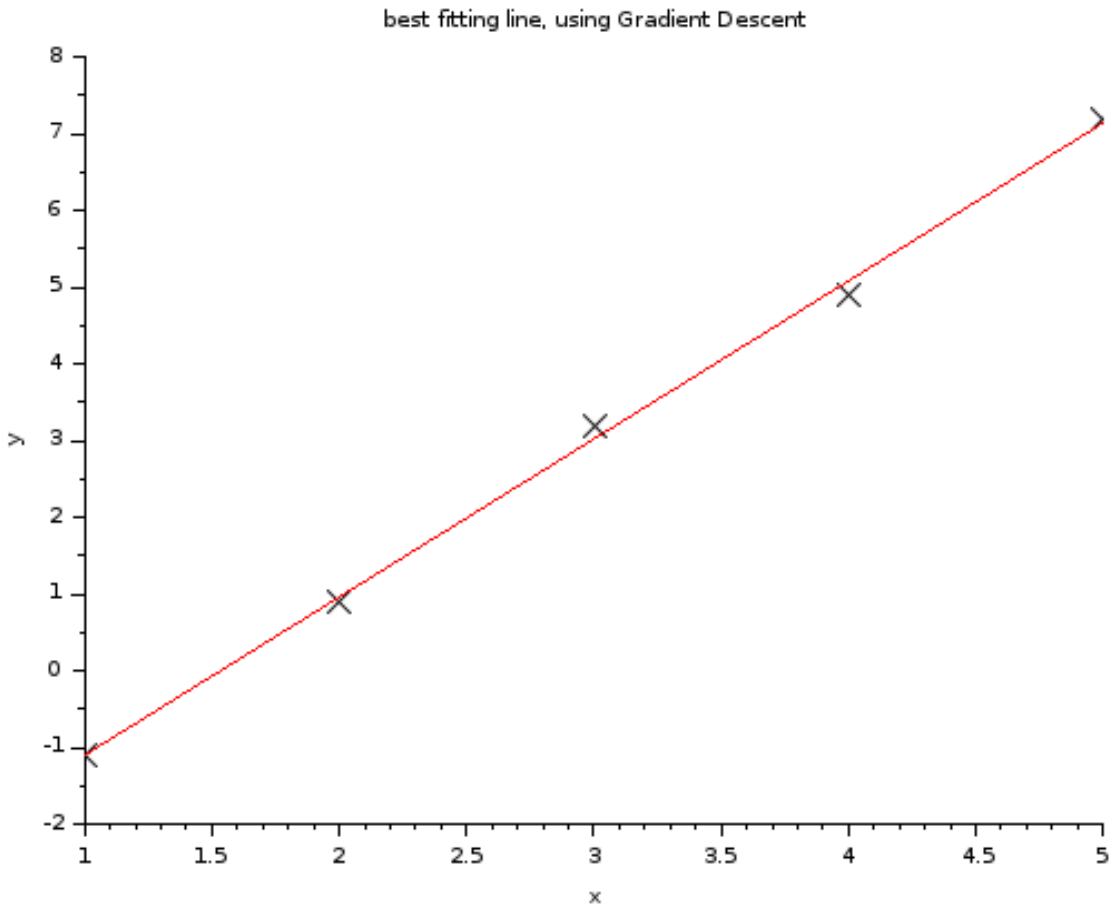


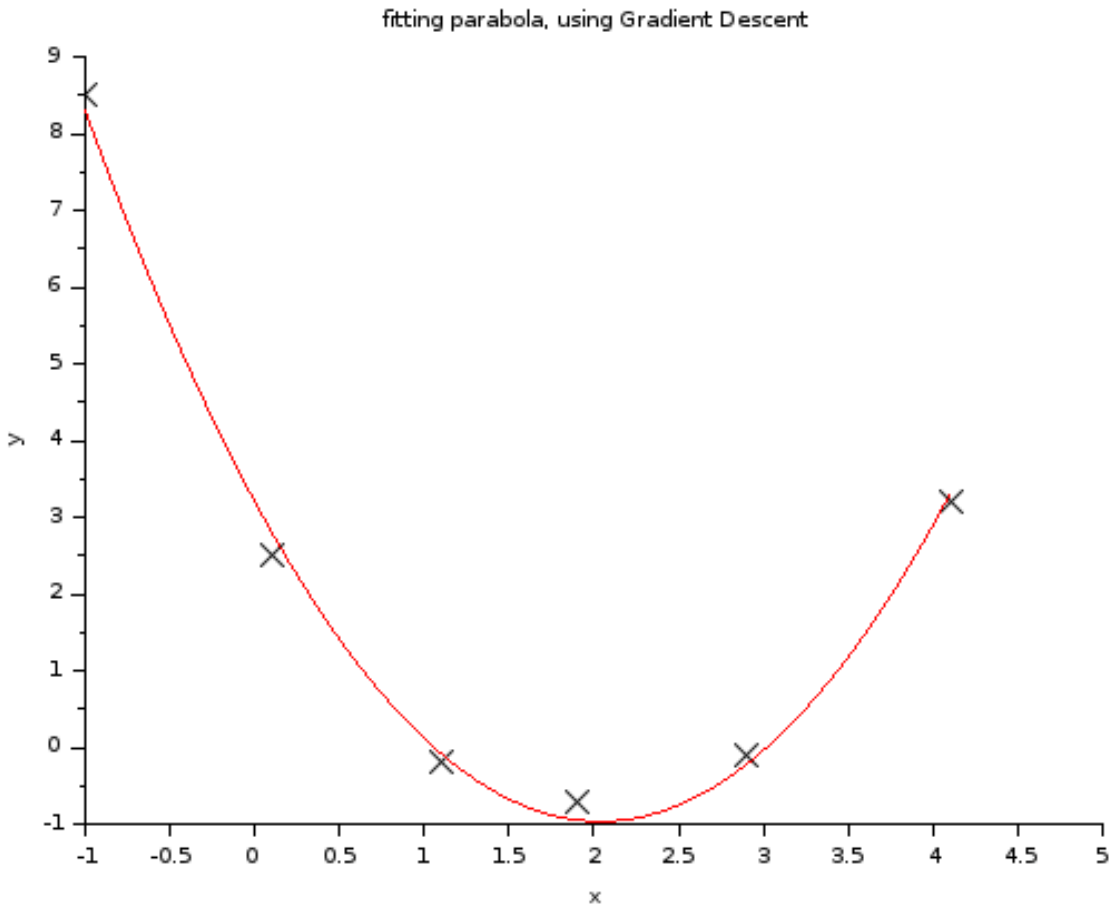

x_0
x_1
x_2
x_3
x_4

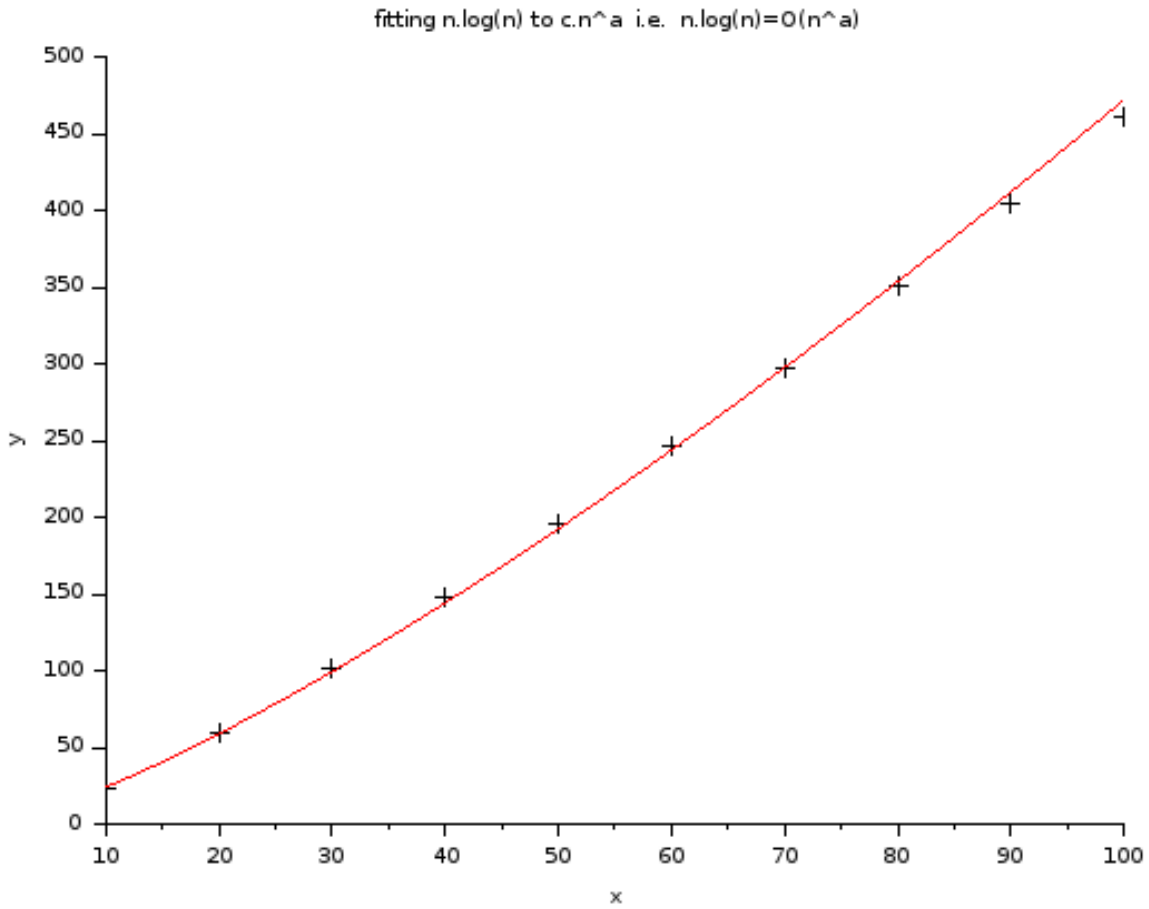


$z = x^2 + y^2$ $z = x^2$ $z = x^2 - y^2$

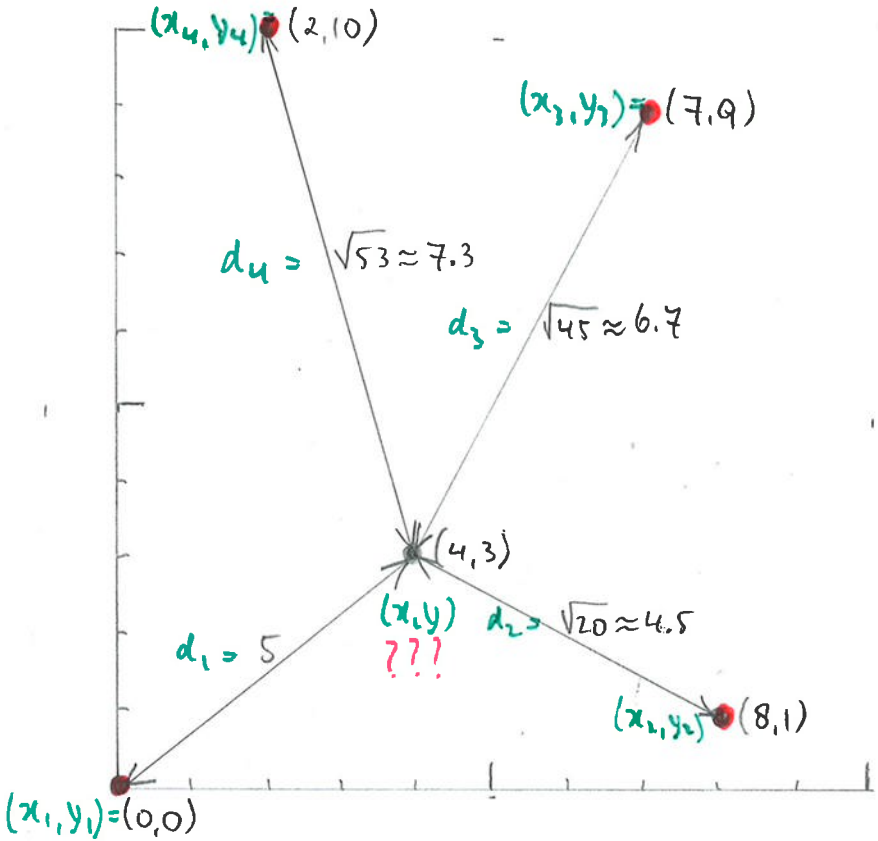
Examples

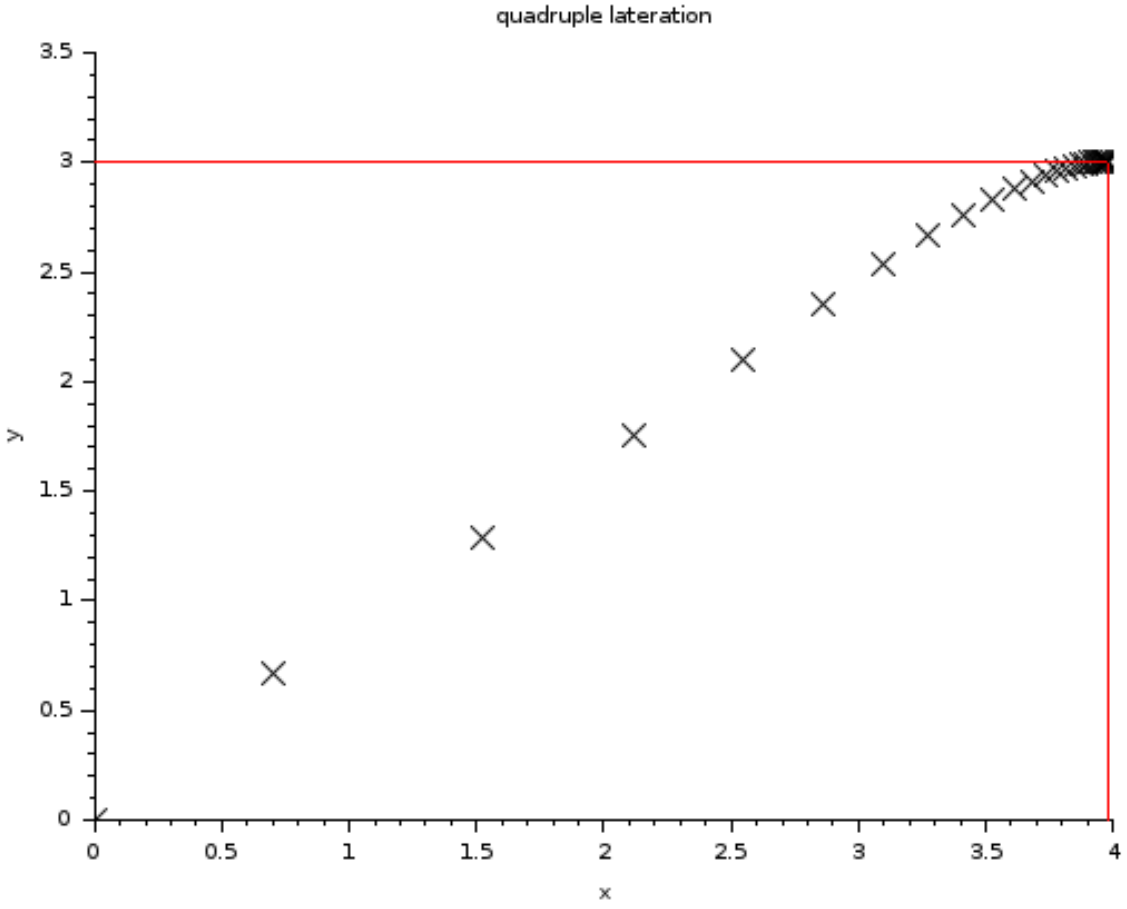

best fitting line, using Gradient Descent
8
7
6
5
4
3
2
1
0
-1
-2
y
1
1.5
2
2.5
3
3.5
4
4.5
5
x


fitting parabola, using Gradient Descent
y
x
9
8
7
6
5
4
3
2
1
0
-1
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5


fitting n.log(n) to c.n^a i.e. n.log(n)=O(n^a)
y
x
0
50
100
150
200
250
300
350
400
450
500
10
20
30
40
50
60
70
80
90
100

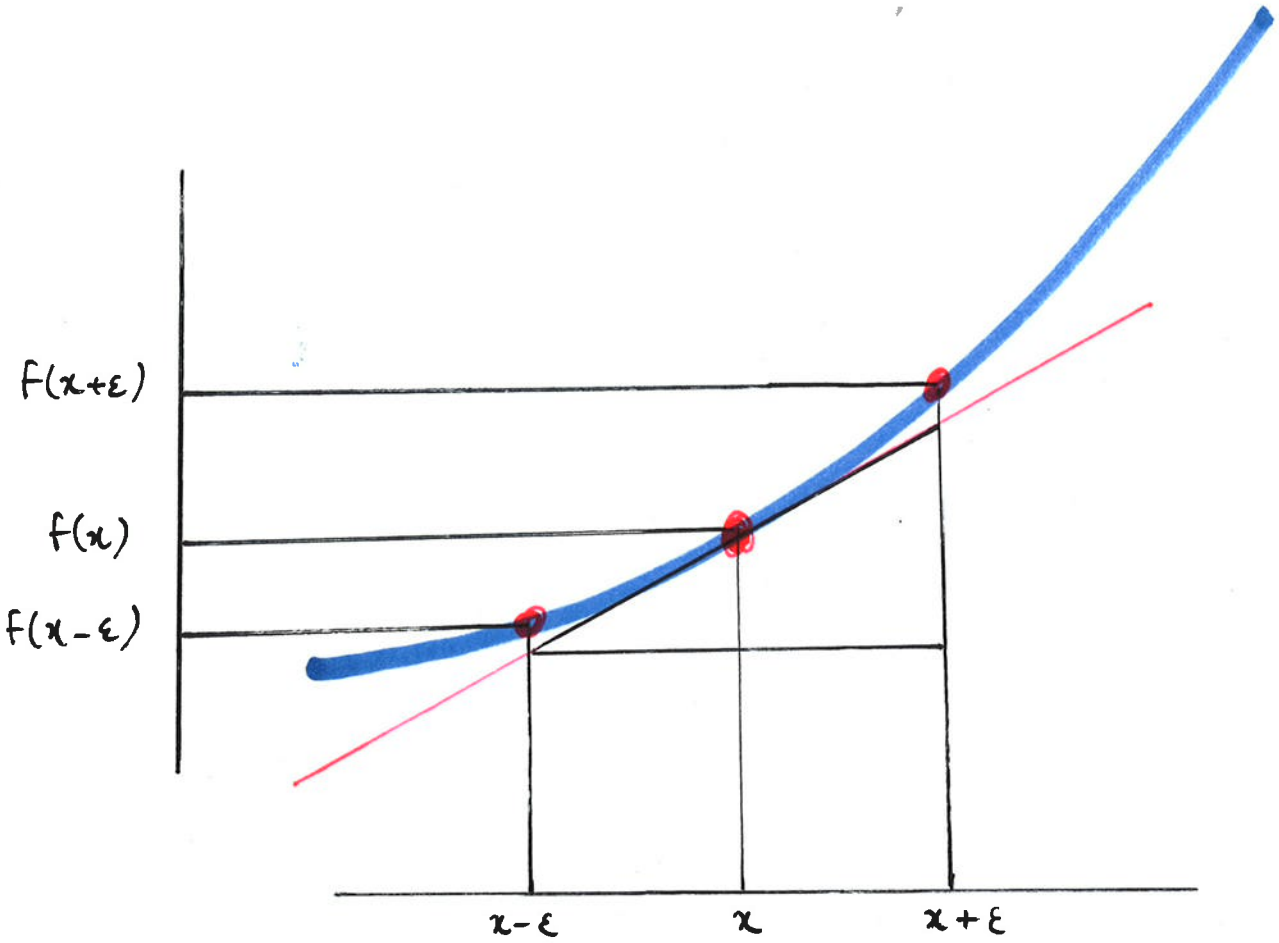
Quadruple Lateration




quadruple lateration
3.5
3
2.5
2
1.5
1
0.5
0
y
0
0.5
1
1.5
2
2.5
3
3.5
4
x

Break

Derivative Approximation


f(x+ε)
f(x)
f(x-ε)
x-ε
x
x+ε

Examples

Conclusion

$$x \mapsto x - \lambda \nabla_f(x)$$

Q & A

End