Education and Culture DG
Lifelong Learning Programme

This project has been funded with support from the European Commission.
This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Describing real life digital pictures with mathematical functions:

Fundacao Escola Profissional de Setubal, Portugal
BHAK Wien 10, Vienna, Austria
Raelingen videregaende skole,Norway

PC based Math
Project for High School Students

1 Description of the Project

Open / Close Print

The objective of the project was to describe real life digital pictures with mathematical functions, so we had to find the best angle for a ski jumper.

We needed to find a picture of a ski jump, which we then had to import into Mathematica. The coordinates of the image are used to calculate the jump. Three jumps were calculated, one too flat, one too high and finally the best one at about 20°, so that the skier lands safely .
Students from Austria, Norway and Portugal were working in this project.

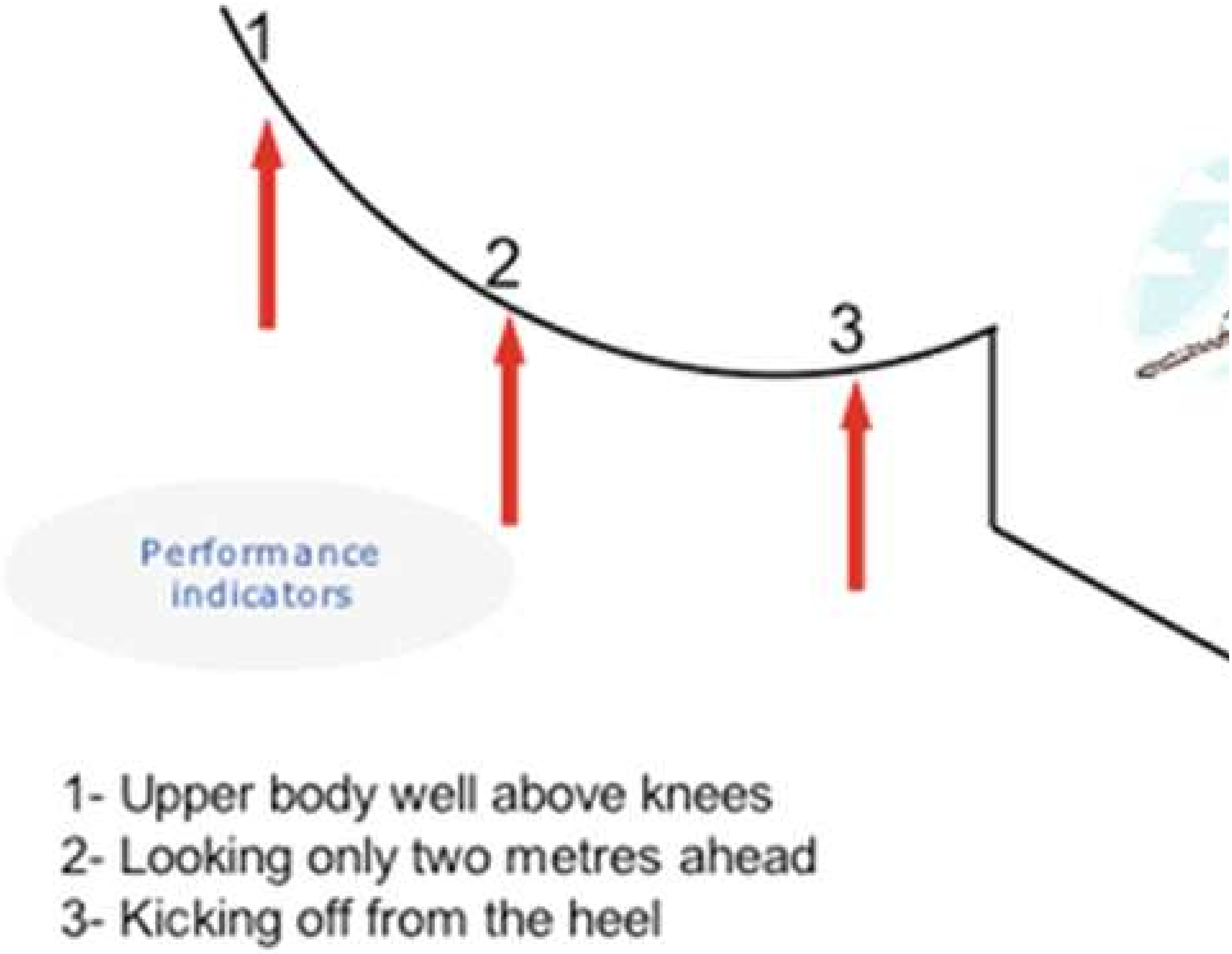
2 Brainstorming and Theory

Open / Close Print

Brainstorming

Open / Close

We are looking for a modell..



What Mathematics do we need
Open / Close

We need "data fitting", a formula for "transposing" our data into real life data, theories of the "differential quotient"," tangent line" and "trigonometric functions".

3 *Developing Models*
Open / Close   Print

Our project work:
Open / Close

This is our "jumping base", we find this picture in the internet.

1. Calculating the ramp.
a) We are calcutating the first part of the ramp by doting the ramp and copiing the datas into the sheet.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.

a)

Input ▹
```
data = {{0.0128571, 0.534286}, {0.0314286, 0.522857},
  {0.0514286, 0.507143}, {0.0728571, 0.491429},
  {0.0957143, 0.48}, {0.124286, 0.462857},
  {0.158571, 0.452857}, {0.181429, 0.444286},
  {0.224286, 0.438571}, {0.252857, 0.434286},
  {0.275714, 0.434286}, {0.294286, 0.434286},
  {0.314286, 0.434286}, {0.322857, 0.435714}}
```

```
{{0.0128571, 0.534286}, {0.0314286, 0.522857},
 {0.0514286, 0.507143}, {0.0728571, 0.491429}, {0.0957143, 0.48},
 {0.124286, 0.462857}, {0.158571, 0.452857}, {0.181429, 0.444286},
 {0.224286, 0.438571}, {0.252857, 0.434286}, {0.275714, 0.434286},
 {0.294286, 0.434286}, {0.314286, 0.434286}, {0.322857, 0.435714}}
```

b)

Input ▹
```
Fit[data, {1, x, x^2}, x]
```

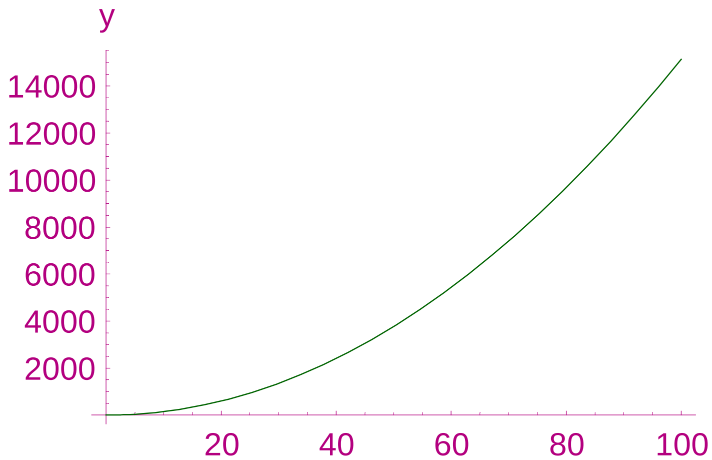
$0.545101 - 0.826548\,x + 1.52204\,x^2$

c)

Input ▹
```
f[x_] =
 0.5451008802514801` - 0.8265478101958164` x + 1.522036209779195` x^2
```

$0.545101 - 0.826548\,x + 1.52204\,x^2$

Input ▹
```
MDPlot[{f[x]}, {x , 0 , 100 } ]
```



- Graphics -

2. Calculating the ramp with real life values
a) Now we have to transpose the datas into reallife values.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.

a)

Input ▹
```
UX = 0.0128571 ; (* it is the first x-value of our data*)
EX = 0.322857; (* it is the maximum x-value of our data*)
UY = 0.534286; (* it is the first y-value of our data*)
EY = 0.434286; (* it is the maximum y-value of our data*)
m = 0; (* it is the starting value of x in  real life*)
n = 50; (* it is the
```

```
   x- value of our maximum in  real life*)
s = 50; (* it is the
   y- value of our maximum in  real life*)
r = 90; (* it is the starting value of y in  real life*)
```

Input ▷

$$\text{datanew1} = \text{Transpose[data]} * \left\{\frac{(n-m)}{(EX-UX)}, \frac{(s-r)}{(EY-UY)}\right\} + \left\{\frac{(m*EX-n*UX)}{(EX-UX)}, \frac{(r*EY-s*UY)}{(EY-UY)}\right\} \text{ // Transpose}$$

```
{{0., 90.}, {2.9954, 85.4284}, {6.22121, 79.1428},
 {9.67742, 72.8572}, {13.3641, 68.2856},
 {17.9724, 61.4284}, {23.5022, 57.4284}, {27.189, 54.},
 {34.1014, 51.714}, {38.7097, 50.}, {42.3963, 50.},
 {45.3918, 50.}, {48.6176, 50.}, {50., 50.5712}}
```

b)

Input ▷ `Fit[datanew1, {1, x, x^2}, x]`

$90.1758 - 1.95278\,x + 0.0234028\,x^2$

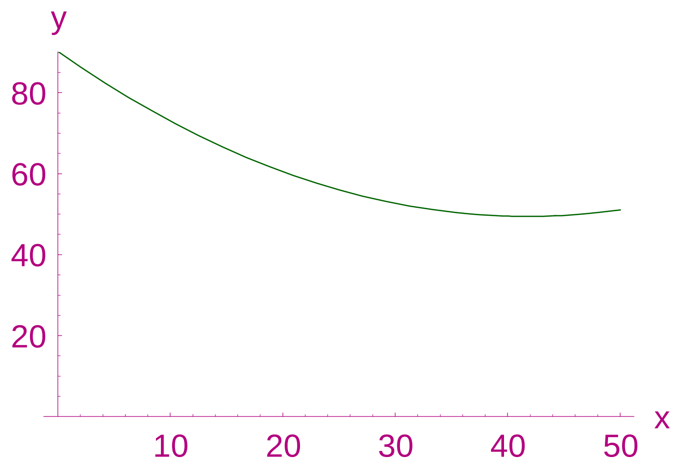
c)

Input ▷

$$\text{ramp1[x\_]} = 90.17578905109345` - 1.9527758394640784`\,x + 0.0234028136629675`\,x^2$$

$90.1758 - 1.95278\,x + 0.0234028\,x^2$

Input ▷ `pic1 = MDPlot[{ramp1[x]}, {x , 0 , 50 }, PlotRange → { 0, 90 } ]`



- Graphics -

--- Now we have the same line as before, but with real life values!! ---

Calculating the landing place.
a) We are calcutating the second part of the ramp by doting the ramp and copiing the datas into the sheet.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.

a)

Input ▷
```
data2 = {{0.327143, 0.408571}, {0.342857, 0.408571},
  {0.365714, 0.408571}, {0.387143, 0.41},
  {0.428571, 0.405714}, {0.482857, 0.392857},
  {0.527143, 0.38}, {0.57, 0.358571}, {0.598571, 0.34},
  {0.631429, 0.317143}, {0.67, 0.288571}, {0.705714, 0.262857},
  {0.744286, 0.238571}, {0.772857, 0.214286},
  {0.818571, 0.182857}, {0.847143, 0.164286},
  {0.871429, 0.144286}, {0.888571, 0.132857}}
```

```
{{0.327143, 0.408571}, {0.342857, 0.408571}, {0.365714, 0.408571},
 {0.387143, 0.41}, {0.428571, 0.405714}, {0.482857, 0.392857},
 {0.527143, 0.38}, {0.57, 0.358571}, {0.598571, 0.34},
 {0.631429, 0.317143}, {0.67, 0.288571}, {0.705714, 0.262857},
 {0.744286, 0.238571}, {0.772857, 0.214286}, {0.818571, 0.182857},
 {0.847143, 0.164286}, {0.871429, 0.144286}, {0.888571, 0.132857}}
```

b)

Input ▷
```
Fit[data2, {1, x, x^2}, x]
```

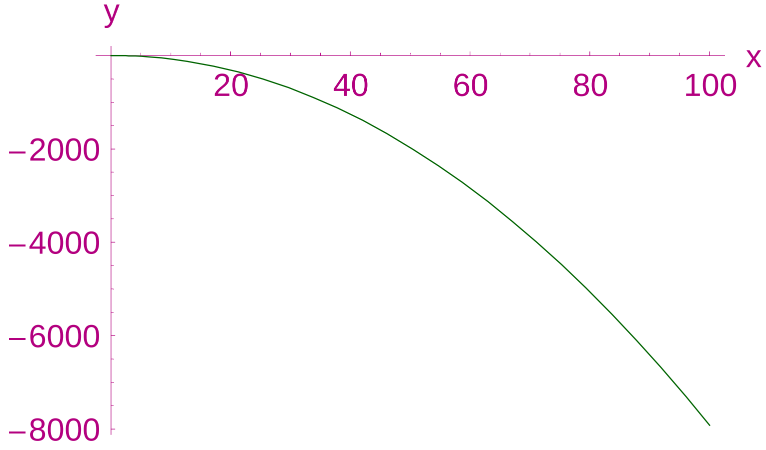
$0.354223 + 0.447571\,x - 0.79673\,x^2$

c)

Input ▷
```
g[x_] = 0.35422343058187944` +
  0.4475711576708544` x - 0.7967296323612434` x^2
```

$0.354223 + 0.447571\,x - 0.79673\,x^2$

Input ▷
```
MDPlot[{g[x]}, {x , 0 , 100 } ]
```



- Graphics -

4. Calculating the landing place with real life values.
a) Now we have to transpose the datas into reallife values.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.

a)

Input ▷
```
UX = 0.327143 ; (* it is the first x-value of our data*)
EX = 0.888571; (* it is the maximum x-value of our data*)
UY = 0.408571; (* it is the first y-value of our data*)
EY = 0.132857; (* it is the maximum y-value of our data*)
m = 50; (* it is the starting value of x in  real life*)
n = 100; (* it is the
   x- value of our maximum in  real life*)
s = 0; (* it is the
   y- value of our maximum in  real life*)
r = 40; (* it is the starting value of y in  real life*)
```

Input ▷

$$\text{datanew2} = \text{Transpose}[\text{daten2}] * \left\{\frac{(n-m)}{(EX-UX)}, \frac{(s-r)}{(EY-UY)}\right\} + \left\{\frac{(m*EX-n*UX)}{(EX-UX)}, \frac{(r*EY-s*UY)}{(EY-UY)}\right\} \text{ // Transpose}$$

{{50., 40.}, {51.3995, 40.}, {53.4351, 40.},
{55.3435, 40.2073}, {59.033, 39.5855}, {63.8677, 37.7202},
{67.8117, 35.855}, {71.6285, 32.7461}, {74.173, 30.0519},
{77.0993, 26.7358}, {80.5344, 22.5907}, {83.715, 18.8601},
{87.1502, 15.3368}, {89.6947, 11.8135}, {93.7659, 7.25389},
{96.3105, 4.55965}, {98.4734, 1.65809}, {100., 3.55271×10^{-15}}}

b)

Input ▹
```
Fit[datanew2, {1, x, x^2}, x]
```

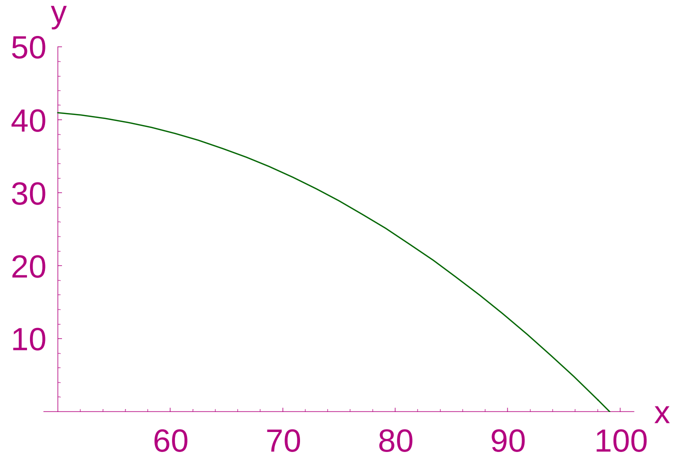
$10.5581 + 1.33725\, x - 0.0145734\, x^2$

c)

Input ▹
```
ramp2[x_] = 10.55806548012172` +
  1.3372503511987441` x - 0.014573379365088452` x^2
```

$10.5581 + 1.33725\, x - 0.0145734\, x^2$

Input ▹
```
pic2 = MDPlot[{ramp2[x]}, {x , 50 , 100 }, PlotRange → { 0, 50 } ]
```

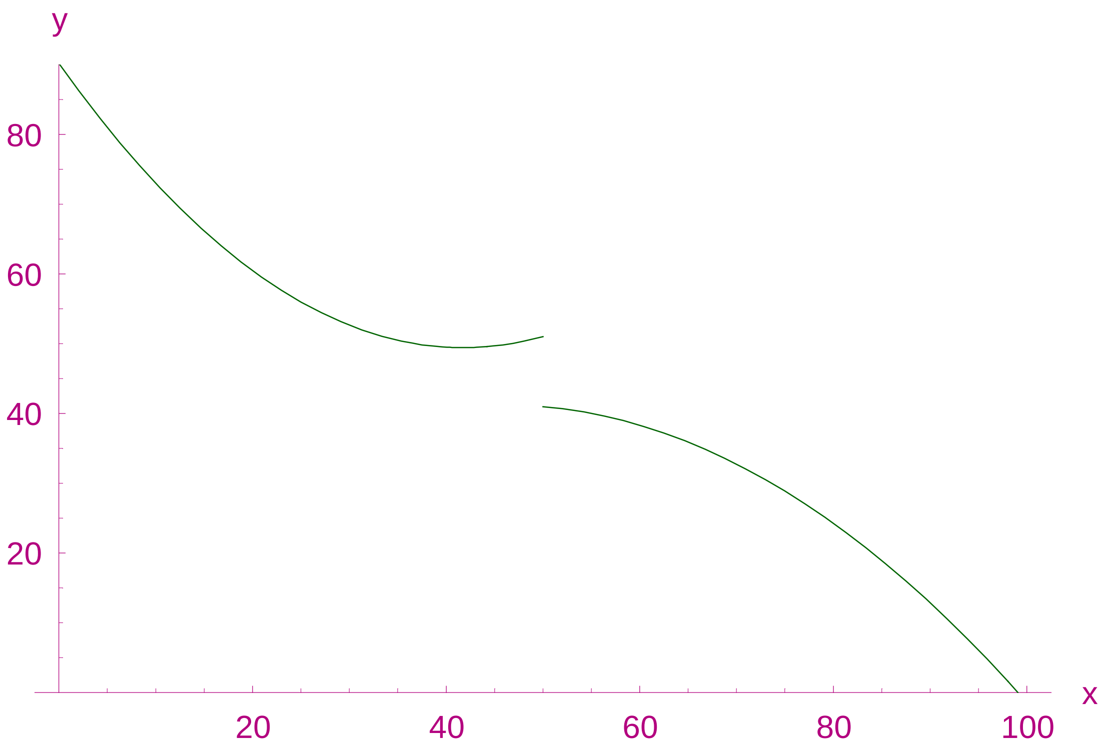


- Graphics -

--- Now we have the same line as before, but with real life values!! ---

5. Now we have transfered the picture into a graphic.

Input ▹
```
Show[ {pic1, pic2 } ]
```



- Graphics -

6. Calculating the first jump.
a) We are calcutating the first jump by doting the flight area and filling in the datas.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.
d) Then we are calculating the inflection point and the maximum point.

a)

Input ▹

```
data3 = {{51.7948, 53.1908}, {53.9908, 55.8558},
  {56.7358, 58.14}, {59.7553, 59.6628}, {64.9708, 61.1856},
  {71.0099, 60.4242}, {77.3234, 56.9979}, {81.441, 49.3838},
  {83.637, 43.2925}, {84.735, 35.2978}, {86.382, 27.303},
  {88.029, 20.0697}, {90.4995, 15.5012}, {90.774, 11.6942},
  {94.617, 7.12576}, {98.7346, 4.46083}, {101.48, 3.31873}}
```

```
{{51.7948, 53.1908}, {53.9908, 55.8558},
 {56.7358, 58.14}, {59.7553, 59.6628}, {64.9708, 61.1856},
 {71.0099, 60.4242}, {77.3234, 56.9979}, {81.441, 49.3838},
 {83.637, 43.2925}, {84.735, 35.2978}, {86.382, 27.303},
 {88.029, 20.0697}, {90.4995, 15.5012}, {90.774, 11.6942},
 {94.617, 7.12576}, {98.7346, 4.46083}, {101.48, 3.31873}}
```

b)

Input ▷
```
Fit[data3, {1, x, x^2, x^3}, x]
```

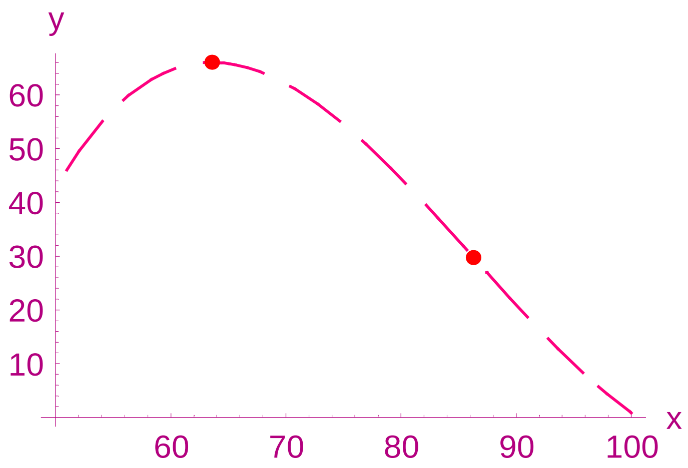
$-758.627 + 32.1996\,x - 0.400886\,x^2 + 0.0015484\,x^3$

c)

Input ▷
```
jump1[x_] = -758.6271311250061` + 32.19962298263877` x -
  0.4008858414757454` x^2 + 0.0015484031553293454` x^3
```

$-758.627 + 32.1996\,x - 0.400886\,x^2 + 0.0015484\,x^3$

Input ▷
```
pic3 = MDPlot[{jump1[x]}, {x , 50 , 100 },
PlotStyle → {{ RGBColor[0.996109, 0, 0.500008], Thickness[.005],
     Dashing[{0.1, 0.05}]}}, Epilog → { Red, PointSize[0.025] ,
    Point[{ 63.58402859168623`, 66.04307755535507`}],
    Point[{ 86.30091805568922`, 29.738562676845277`}] }]
```



- Graphics -

d)

Input ▷
```
 → Pure NSolve ;
Clear[x];
MDRealOnly[NSolve[{ jump1'[x] == 0}, { x } ]]
```

{{x → 63.584}, {x → 109.018}}

Input ▷
```
jump1[63.58402859168623`]
```

66.0431

Input ▷
```
 → Pure NSolve ;
Clear[x];
MDRealOnly[NSolve[{ jump1''[x] == 0}, { x } ]]
```

{{x → 86.3009}}

Input ▷
```
jump1[86.30091805568922`]
```

29.7386

7. Calculating the second jump.
a) We are calcutating the second jump by doting the flight area and filling in the datas.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.
d) Then we are calculating the inflection point and the maximum point.

a)

Input ▹
```
data4 = {{50.6126, 51.7203}, {53.3542, 52.861}, {57.1924, 54.0016},
  {62.4015, 53.6214}, {67.6105, 50.9598}, {71.1746, 46.0168},
  {75.0128, 38.7925}, {76.3836, 34.6099}, {78.3028, 29.2867},
  {82.4152, 24.724}, {87.0759, 17.4996}, {90.64, 12.5567}}
```

```
{{50.6126, 51.7203}, {53.3542, 52.861}, {57.1924, 54.0016},
 {62.4015, 53.6214}, {67.6105, 50.9598}, {71.1746, 46.0168},
 {75.0128, 38.7925}, {76.3836, 34.6099}, {78.3028, 29.2867},
 {82.4152, 24.724}, {87.0759, 17.4996}, {90.64, 12.5567}}
```

b)

Input ▹
```
Fit[data4, {1, x, x^2, x^3}, x]
```

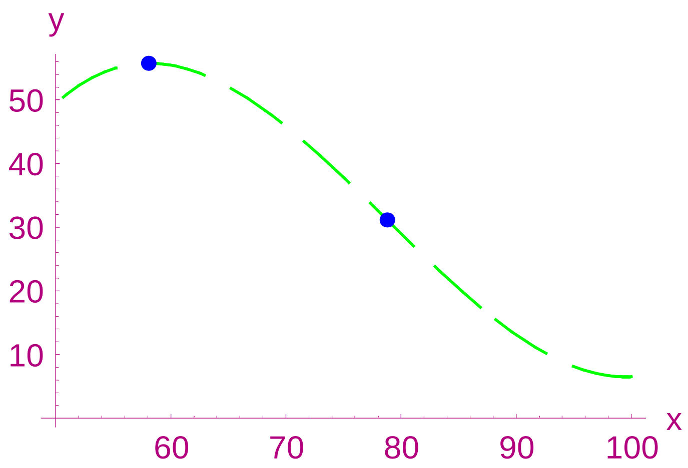
$-506.872 + 24.0513\,x - 0.327876\,x^2 + 0.00138701\,x^3$

c)

Input ▹
```
jump2[x_] = -506.8715801347884` + 24.05134653144517` x -
  0.3278755656826356` x^2 + 0.0013870123316435188` x^3
```

$-506.872 + 24.0513\,x - 0.327876\,x^2 + 0.00138701\,x^3$

Input ▹
```
pic4 = MDPlot[{jump2[x]}, {x , 50 , 100 },
PlotStyle → {{ Green, Thickness[.005], Dashing[{0.1, 0.05}]} },
  Epilog → { Blue, PointSize[0.025] ,
    Point[{58.0898067114866 , 55.756561704413116 }],
    Point[{ 78.79659952124688, 31.12744526230813 }] } ]
```



- Graphics -

d)

→ Pure NSolve;

Input ▹
```
Clear[x];
MDRealOnly[NSolve[{ jump2'[x] == 0}, { x } ]]
```

{{x → 58.0898}, {x → 99.5034}}

Input ▹
```
jump2[58.0898067114866]
```

55.7566

→ Pure NSolve;

Input ▹
```
Clear[x];
MDRealOnly[NSolve[{ jump2''[x] == 0}, { x } ]]
```

{{x → 78.7966}}

Input ▹
```
jump2[78.79659952124688]
```

31.1274

8.

Calculating the third jump.
a) We are calcutating the third jump by doting the flight area and filling in the datas.
b) We have to fit the datas to define a line.
c) Afterwards we define the line and plot it.
d) Then we are calculating the inflection point and the maximum point.

a)

```
data5 = {{53.4418, 54.7136}, {57.8338, 56.2365},
  {61.9513, 54.7136}, {65.7944, 49.3838}, {70.1864, 42.1504},
```

```
{74.8529, 34.9171}, {78.1469, 29.5872}, {81.99, 24.2574},
{86.1075, 19.3083}, {87.48, 17.7854}, {88.578, 16.6433},
{89.676, 15.1205}, {93.519, 10.5521}, {97.9111, 10.5521}}
```

```
{{53.4418, 54.7136}, {57.8338, 56.2365},
 {61.9513, 54.7136}, {65.7944, 49.3838}, {70.1864, 42.1504},
 {74.8529, 34.9171}, {78.1469, 29.5872}, {81.99, 24.2574},
 {86.1075, 19.3083}, {87.48, 17.7854}, {88.578, 16.6433},
 {89.676, 15.1205}, {93.519, 10.5521}, {97.9111, 10.5521}}
```

b)

Input ▷
```
Fit[data5, {1, x, x^2, x^3}, x]
```

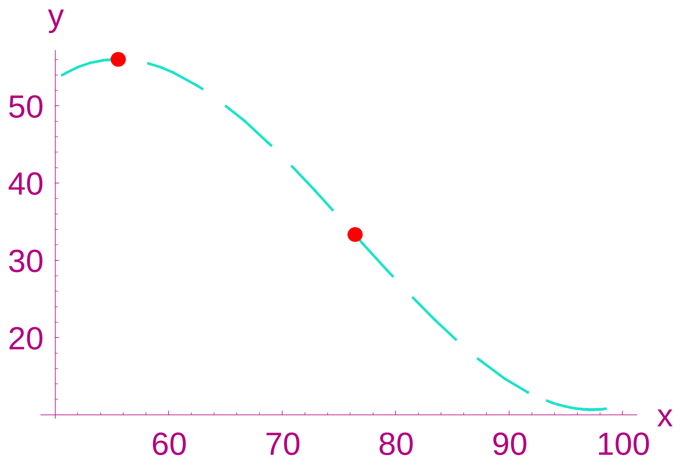
$-398.179 + 20.2036\,x - 0.285772\,x^2 + 0.00124681\,x^3$

c)

Input ▷
```
jump3[x_] = -398.1793759697596` + 20.203588313157784` x -
  0.28577228053239695` x^2 + 0.0012468109137881607` x^3
```

$-398.179 + 20.2036\,x - 0.285772\,x^2 + 0.00124681\,x^3$

Input ▷
```
pic5 = MDPlot[{jump3[x]}, {x , 50 , 100 },
PlotStyle → {{ RGBColor[0.109377, 0.886732, 0.792981],
     Thickness[.005], Dashing[{0.1, 0.05}]} },
  Epilog → { Red, PointSize[0.025] ,
    Point[{ 55.527731507398265`, 56.01692547371684` }],
    Point[{ 76.4008606202472`, 33.339525015597246` }] }]
```



- Graphics -

d)

→ **Pure NSolve ;**

Input ▷
```
Clear[x];
MDRealOnly[NSolve[{ jump3'[x] == 0}, { x } ]]
```

```
{{x → 55.5277}, {x → 97.274}}
```

Input ▹
```
jump3[55.527731507398265]
```

```
56.0169
```

→ Pure NSolve ;

Input ▹
```
Clear[x];
MDRealOnly[NSolve[{ jump3''[x] == 0}, { x } ]]
```

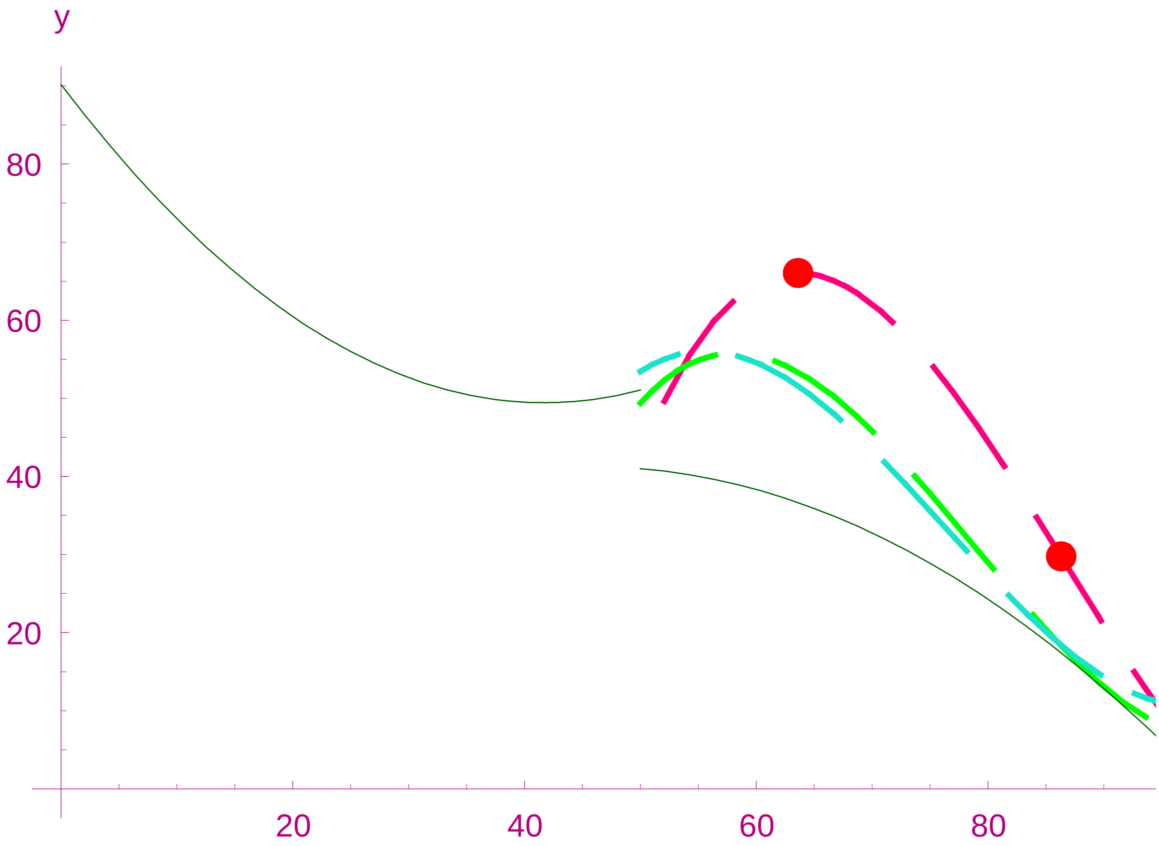
```
{{x → 76.4009}}
```

Input ▹
```
jump3[76.4008606202472]
```

```
33.3395
```

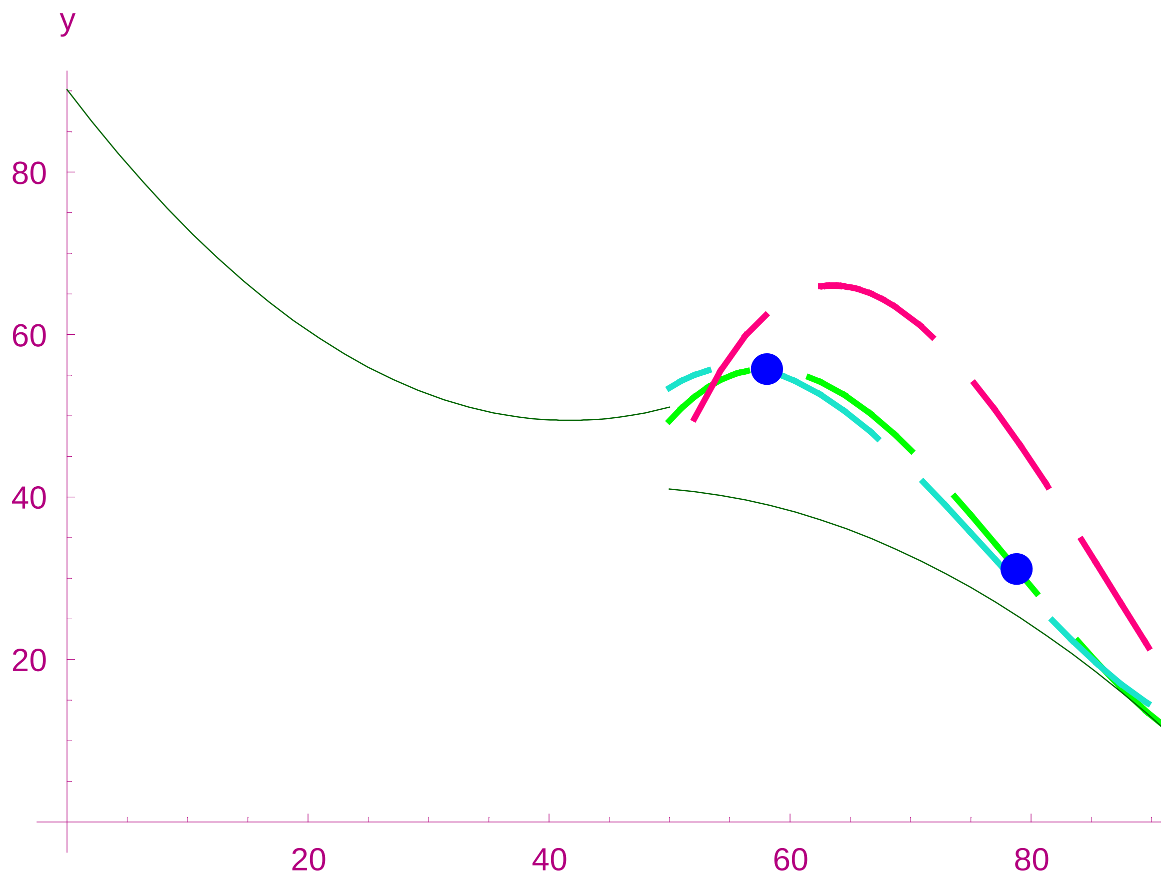
9. Now we are showing all the jumps, including the inflection- and maximum point.

Input ▹
```
Show[pic3, pic4, pic5, pic2, pic1]
```



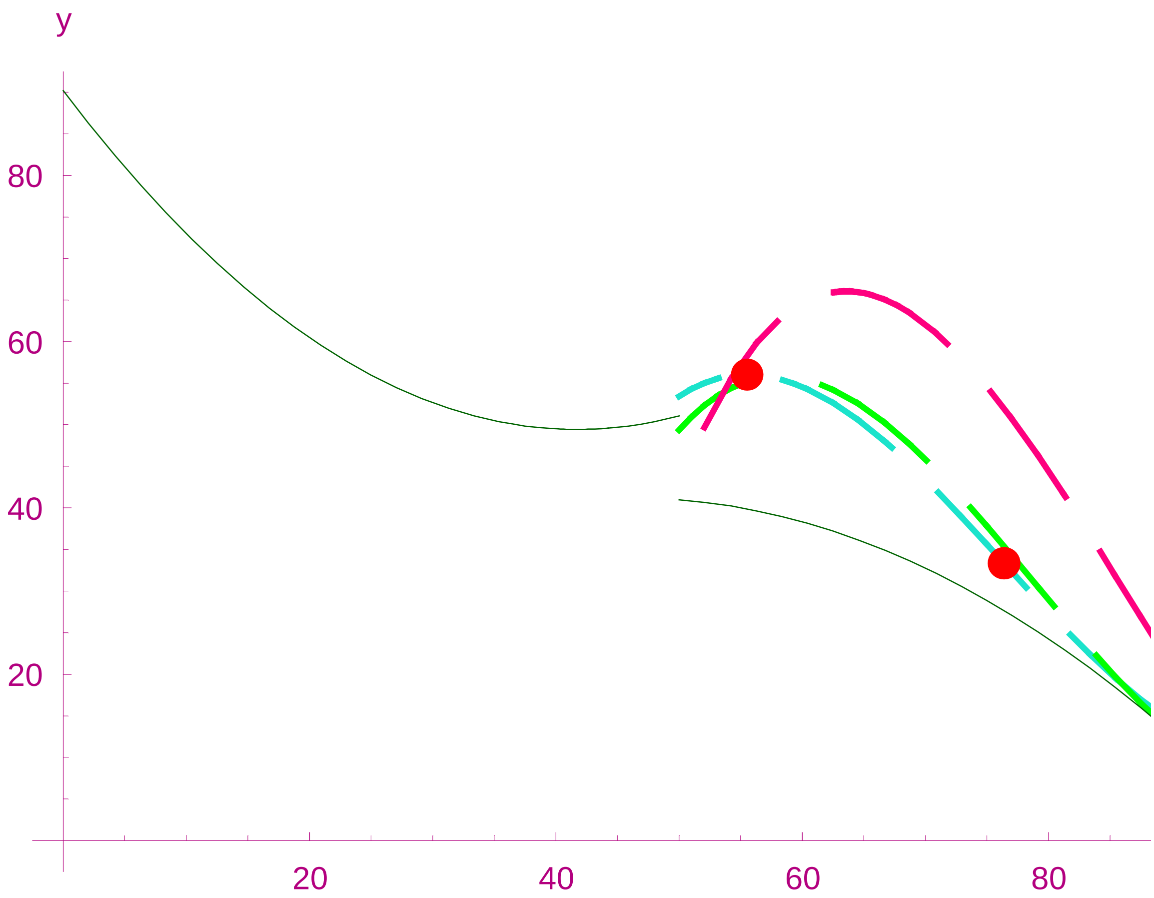

```
- Graphics -
```

Input ▹
```
Show[pic4, pic3, pic5, pic2, pic1]
```



- Graphics -

Input ▹ **Show[pic5, pic4, pic3, pic2, pic1]**



- Graphics -

10. Calculating the tangente line
a) We have to find the x-coordinate by clicking into the the graph
Just hold Crtl and take the last part of the ramp (ramp1).
b) Then we calculate the slope.
c) Afterwards we define the line and plot it.
d) Put together all line + the tangente line.
e) Finding the angel.

a)

Input ▷
```
ramp1'[50]
```

```
0.387506
```

b)

```
→ Pure NSolve;
```

Input ▷
```
Clear[x];
MDRealOnly[NSolve[{50 == 50 * 0.3875055268326715` + d}, { d } ]]
```

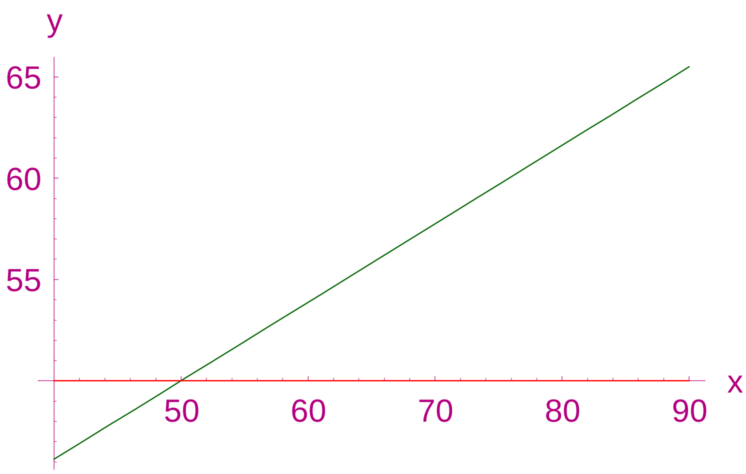
```
{{d → 30.6247}}
```

c)

Input ▷ `wt[x_] = 0.3875055268326715 * x + 30.624723658366428``

`30.6247 + 0.387506 x`

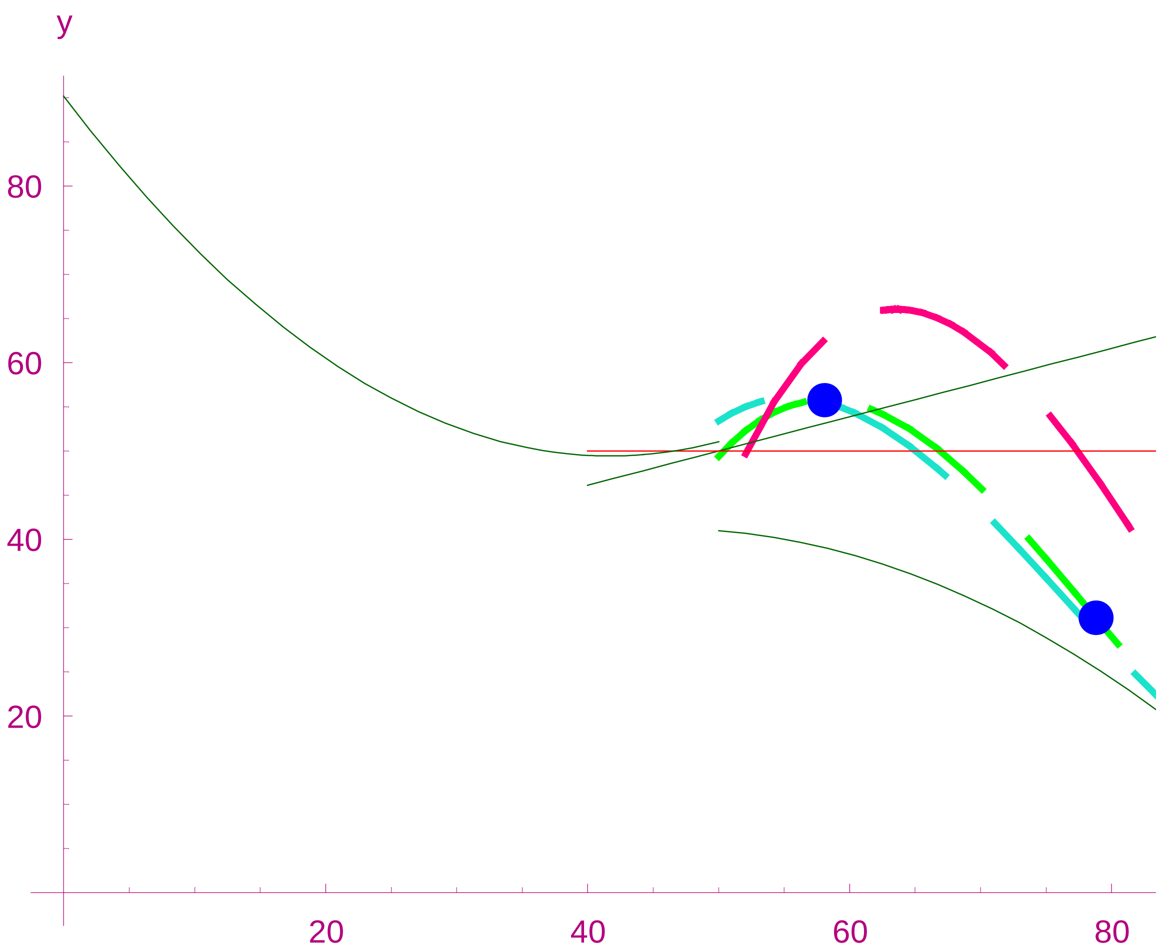
Input ▷ `pic6 = MDPlot[{wt[x], 50}, {x , 40, 90 } ]`



`- Graphics -`

d)

Input ▷ `Show[pic4, pic5, pic3, pic6, pic2, pic1]`



- Graphics -

e)

Input ▹
```
m1 = 0;
m2 = 0.3875055268326715;
φ = ArcTan[Abs[(m2 - m1)/(1 + m2 * m1)]] / ° // N
```

```
21.1816
```

The skier takes of with a slope of 21.1816°.

4 Result and Summary

Open/Close Print

Final remarks:

Open / Close

The ski jumper will make the best jump, when he will take off under an angel of 21°.

5 Our Team

Open/Close Print

Participants

Open / Close

Two groups of students from Austria, Norway and Portugal have worked on this project. The following two pictures show the two project groups.

PC based Math Project for High School Students

This picture was taken from http://lyska.net/MathProject/index.html

Our Experience with the Project

Open / Close

This project gave us a view outside of our country. We find new friends in whole europe and it was a great experience for us to work with other students from foreign countries in a mathematical project.

INTERNET W3 Open / Close Print

List of our sources:

PCMLogo: [PCM Homepage]