```
>
```

October 2015

Nonlinear Regression with Maple

Univ.-Prof. Dr.-Ing. habil. Josef *BETTEN*
RWTH Aachen University
Mathematical Models in Materials Science and Continuum Mechanics
Augustinerbach 4-20
D-52056 A a c h e n , Germany

<betten@mmw.rwth-aachen.de>

Abstract

Many Authors have discussed *Nonlinear Regression* based upon the *LEVENBERG-MARQUART algorithm.* Often the *SIGMAPLOT* Program is very useful, too.

In this worksheet the Maple Program *NonlinearFit [Statistics]* has been prefered, to fit nonlinear model functions to given data. This program is very effectvely and simple to use. In the following some examples have been applied, e.g., the *envelopes* of *FRENEL's integrals* or the *hardening* of an *aluminium alloy* etc.

FRENEL's Integrals

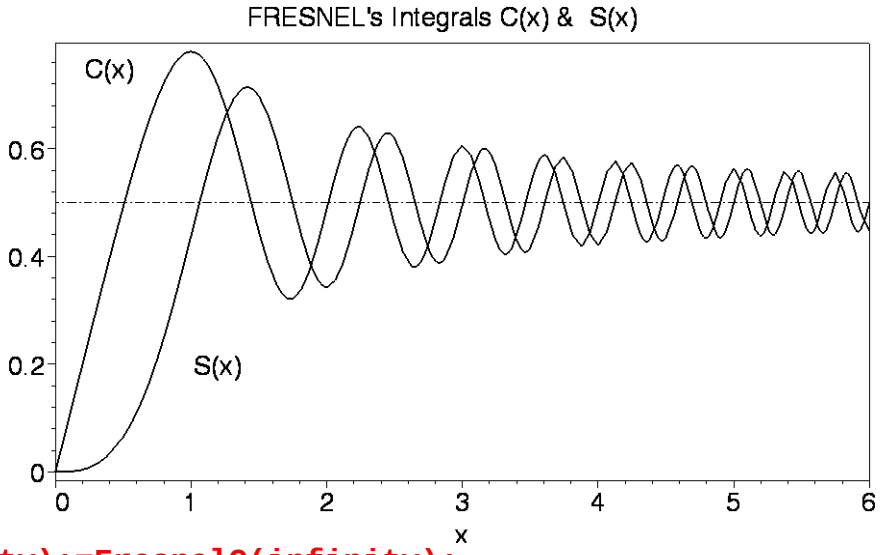
```
> restart:
> C(x):=Int(cos(Pi*t/2),t=0..x)=FresnelC(x);
```

$$C(x) := \int_0^x \cos\left(\frac{\pi\, t}{2}\right) dt = \mathrm{FresnelC}(x)$$

```
> S(x):=Int(sin(Pi*t/2),t=0..x)=FresnelS(x);
```

$$S(x) := \int_0^x \sin\left(\frac{\pi\, t}{2}\right) dt = \mathrm{FresnelS}(x)$$

```
> alias(th=thickness,co=color):
> p[1]:=plot({FresnelC(x),FresnelS(x)},x=0..6,
  axes=boxed,th=2,co=black):
> p[2]:=plot(1/2,x=0..6,linestyle=4,co=black,
  title="FRESNEL's Integrals C(x) &  S(x)"):
> p[3]:=plots[textplot]({[0.4,0.75,`C(x)`],[1.2,0.2,`S(x)`]}):
> plots[display](seq(p[k],k=1..3));
```




```
> C(infinity):=FresnelC(infinity);
  S(infinity):=FresnelS(infinity);
```

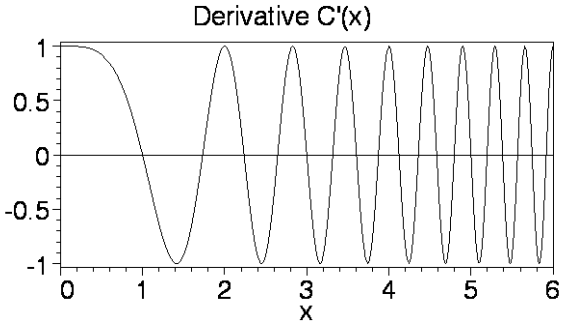
$$C(\infty) := \frac{1}{2}$$

$$S(\infty) := \frac{1}{2}$$

```
> # coordinates of the extrema from `C'`(x)=0:
> `C'`(x):=Diff(FresnelC(xi),xi)=diff(FresnelC(x),x);
```

$$C'(x) := \frac{d}{d\xi}\,\mathrm{FresnelC}(\xi) = \cos\left(\frac{\pi x^2}{2}\right)$$

```
> plot({0,rhs(`C'`(x))},x=0..6,axes=boxed,co=black,
  title="Derivative C'(x)");
```




```
> # From C'(x)=cos(Pi*x^2/2)=0 we arrive at:
> X[extrema][i]:=sqrt(2*i-1);
```

$$X_{extrema_i} := \sqrt{2\,i - 1}$$

```
> for i from 1 to 20 do X[extrema][i]:=sqrt(2*i-1) od:
> with(LinearAlgebra):
> X[extrem]:=vector([seq(X[extrema][i],i=1..20)]);
```

$$X_{extrem} := [1, \sqrt{3}, \sqrt{5}, \sqrt{7}, 3, \sqrt{11}, \sqrt{13}, \sqrt{15}, \sqrt{17}, \sqrt{19}, \sqrt{21}, \sqrt{23}, 5, 3\sqrt{3}, \sqrt{29}, \sqrt{31}, \sqrt{33}, \sqrt{35}, \sqrt{37}, \sqrt{39}]$$

```
> x[max]:=vector([seq(X[extrema][i],i=1..20,2)]);
```

$$x_{max} := [1, \sqrt{5}, 3, \sqrt{13}, \sqrt{17}, \sqrt{21}, 5, \sqrt{29}, \sqrt{33}, \sqrt{37}]$$

```
> x[max]:=evalf(%,4);
```

$$x_{max} := [1., 2.236, 3., 3.606, 4.123, 4.583, 5., 5.385, 5.745, 6.083]$$

```
> x[min]:=vector([seq(X[extrema][i],i=2..20,2)]);
```

$$x_{min} := [\sqrt{3}, \sqrt{7}, \sqrt{11}, \sqrt{15}, \sqrt{19}, \sqrt{23}, 3\sqrt{3}, \sqrt{31}, \sqrt{35}, \sqrt{39}]$$

```
> x[min]:=evalf(%,4);
```

$$x_{min} := [1.732, 2.646, 3.317, 3.873, 4.359, 4.796, 5.196, 5.568, 5.916, 6.245]$$

Generating envelopes for FresnelC(x) utilyzing Statistics[NonlinearFit] in Maple

```
> restart:  with(Statistics):
> x[max][i]:=sqrt(4*i-3);
```

$$x_{max_i} := \sqrt{4\,i - 3}$$

```
> X:=evalf(vector([seq(sqrt(4*i-3),i=1..10)]),4);
  # x-koordinates of the maxima:
```

$$X := [1., 2.236, 3., 3.606, 4.123, 4.583, 5., 5.385, 5.745, 6.083]$$

```
> whattype(X);
```

symbol

```
> for i from 1 to 10 do
  C[max][i]:=evalf(subs(x=sqrt(4*i-3),FresnelC(x)),4) od:
> Y:=evalf(vector([seq(C[max][i],i=1..10)]));
  # y-coordinates of the maxima:
```

$$Y := [0.7799, 0.6408, 0.6057, 0.5881, 0.5771, 0.5694, 0.5636, 0.5591, 0.5554, 0.5523]$$

```
> whattype(Y);
```

symbol

```
> DATA:=seq([X[i],Y[i]],i=1..10);
```

$$DATA := [1., 0.7799], [2.236, 0.6408], [3., 0.6057], [3.606, 0.5881], [4.123, 0.5771],$$
$$[4.583, 0.5694], [5., 0.5636], [5.385, 0.5591], [5.745, 0.5554], [6.083, 0.5523]$$

For the upper envelope y[max](x) a "four parameters double exponential decay" is a suitable nonlinear model function to the calculated DATA:

```
> f:=(t,a,b,c,d) -> 1/2+a*exp(-b*t)+c*t*exp(-d*t);
```

$$f := (t, a, b, c, d) \rightarrow \frac{1}{2} + a\,\mathbf{e}^{(-b\,t)} + c\,t\,\mathbf{e}^{(-d\,t)}$$

```
> y[max]:=unapply(evalf(NonlinearFit(f(t,a,b,c,d),X,Y,t),4),t);
```

$$y_{max} := t \rightarrow 0.5000 + 0.5824\,\mathbf{e}^{(-0.8913\,t)} + 0.05643\,t\,\mathbf{e}^{(-0.3191\,t)}$$

```
> y[max](x):=subs(t=x,y[max](t));
```

$$y_{max}(x) := 0.5000 + 0.5824\,\mathbf{e}^{(-0.8913\,x)} + 0.05643\,x\,\mathbf{e}^{(-0.3191\,x)}$$

```
> y[max](0):=evalf(subs(x=0,y[max](x)),4);
```

$$y_{max}(0) := 1.082$$

```
> y[max](infinity):=evalf(subs(x=100000,y[max](x)),4);
```

$$y_{max}(\infty) := 0.5000$$

```
> FresnelC(0):=simplify(subs(x=0,FresnelC(x)));
```

$$\mathrm{FresnelC}(0) := 0$$

```
> FresnelC(infinity):=evalf(subs(x=infinity,FresnelC(x)),2);
```

$$\mathrm{FresnelC}(\infty) := 0.5$$

```
> # lower envelope y[min](x):
> x[min][k]:=sqrt(4*k-1);
```

$$x_{min_k} := \sqrt{4\,k - 1}$$

```
> P:=evalf(array([seq(sqrt(4*k-1),k=1..10)]),4);
  # x-coordinates of the minima:
```

$$P := [1.732, 2.646, 3.317, 3.873, 4.359, 4.796, 5.196, 5.568, 5.916, 6.245]$$

```
> whattype(P);
```

symbol

```
> for i from 1 to 10 do
  C[min][i]:=evalf(subs(x=sqrt(4*i-1),FresnelC(x)),4) od:
  Q:=evalf(array([seq(C[min][i],i=1..10)]));
  #  y-coordinates of the minima:
```

$$Q := [0.3211, 0.3804, 0.4043, 0.4179, 0.4270, 0.4337, 0.4388, 0.4428, 0.4462, 0.4490]$$

```
> whattype(Q);
```

symbol

```
> data:=seq([P[i],Q[i]],i=1..10);
```

$$data := [1.732, 0.3211], [2.646, 0.3804], [3.317, 0.4043], [3.873, 0.4179], [4.359, 0.4270],$$
$$[4.796, 0.4337], [5.196, 0.4388], [5.568, 0.4428], [5.916, 0.4462], [6.245, 0.4490]$$

For the lower envelope y[min](x) a "three parameters exponential decay" is a suitable nonlinear model function to the calculated data:

```
> g:=(t,p,q,r) -> (1/2)*(1-exp(-p*t))+q*t*exp(-r*t);
```

$$g := (t, p, q, r) \to \frac{1}{2} - \frac{1}{2}\,\mathbf{e}^{(-p\,t)} + q\,t\,\mathbf{e}^{(-r\,t)}$$

```
> y[min]:=unapply(evalf(NonlinearFit(g(t,p,q,r),P,Q,t),4),t);
```

$$y_{min} := t \to 0.5000 - 0.5000\,\mathbf{e}^{(-0.7611\,t)} - 0.04185\,t\,\mathbf{e}^{(-0.2762\,t)}$$

```
> y[min](x):=1/2-(1/2)*exp(-0.7611*x)-0.04185*x*exp(-0.2762*x);
```

$$y_{min}(x) := \frac{1}{2} - \frac{1}{2}\,\mathbf{e}^{(-0.7611\,x)} - 0.04185\,x\,\mathbf{e}^{(-0.2762\,x)}$$

```
> y[min](0):=convert(evalf(subs(x=0,y[min](x))),rational);
```

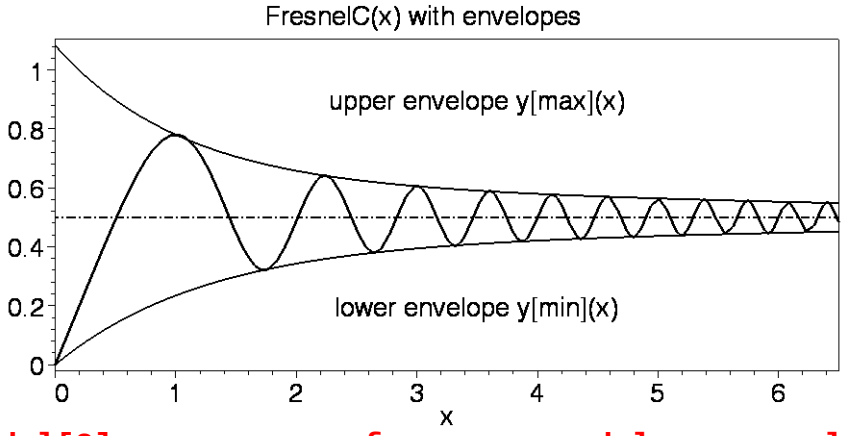
$$y_{min}(0) := 0$$

```
> y[min](infinity):=convert(limit(y[min](x),x=infinity),rational);
```

$$y_{min}(\infty) := \frac{1}{2}$$

```
> # The boundary conditions are fulfilled.
```

```
> alias(th=thickness,co=color):
> p[1]:=plot({y[min](x),y[max](x)},x=0..6.5,th=2,co=black):
> p[2]:=plot(FresnelC(x),x=0..6.5,th=3,co=black,axes=boxed):
> p[3]:=plot(1/2,x=0..6.5,linestyle=4,th=2,co=black),
  title="FresnelC(x) with envelopes":
> p[4]:=plots[textplot]({[3.5,0.9,`upper envelope y[max](x)`],
  [3.5,0.2,`lower envelope y[min](x)`]}):
> plots[display](seq(p[k],k=1..4));
```




```
> # L[2] and l[2] error norms for upper and lower envelopes,
  respectaviely:
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected

> for i from 1 to 10 do
  v[i]:=evalf(subs(x=DATA[i][1],y[max](x))-DATA[i][2],4) od:
> V:=vector([seq(v[i],i=1..10)]);
```

$$V := [-0.00009, 0.00041, -0.00054, -0.00032, 0.00007, 0.00034, 0.00036, 0.00021, -0.00008, -0.00043]$$

```
> L[2]:=(1/sqrt(number_of_points))*Norm(V,2)=
  evalf((1/sqrt(10))*norm(V,2),4);
```

$$L_2 := \frac{\mathrm{Norm}(V, 2)}{\sqrt{number\_of\_points}} = 0.0003251$$

```
> L[infinity]:=Norm(V,infinity)=evalf(norm(V,infinity),4);
```

$$L_\infty := \mathrm{Norm}(V, \infty) = 0.00054$$

```
> L[infinity]:=Max(abs(v[1..10]))=evalf(abs(v[3]),4);
```

$$L_\infty := \mathrm{Max}(|v_{1..10}|) = 0.00054$$

```
> for i from 1 to 10 do
  w[i]:=evalf(subs(x=data[i][1],y[min](x))-data[i][2],4) od:
> W:=vector([seq(w[i],i=1..10)]);
```

$$W := [0.00018, -0.00045, 0.00015, 0.00028, 0.00016, -0.00005, -0.00017, -0.00007, -0.00005, 0.00011]$$

```
> l[2]:=(1/sqrt(number_of_points))*Norm(W,2)=
  evalf((1/sqrt(10))*norm(W,2),4);
```

$$l_2 := \frac{\mathrm{Norm}(W, 2)}{\sqrt{number\_of\_points}} = 0.0002030$$

```
> l[infinity]:=Norm(W,infinity)=
  evalf((1/(10)^(1/infinity))*norm(W,infinity),4);
```

$$l_\infty := \mathrm{Norm}(W, \infty) = 0.00045$$

```
> l[infinity]:=Max(abs(w[1..10]))=evalf(abs(w[2]),4);
```

$$l_\infty := \mathrm{Max}(|w_{1\,..\,10}|) = 0.00045$$

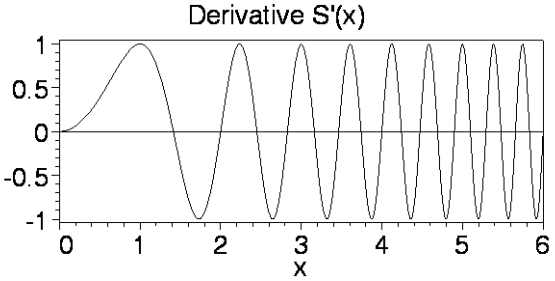
The *error norms* above show that both the upper and lower envelopes are suitable nonlinear model functions to both DATA and data.

Envelopes for FresnelS(x):

```
> restart:
> # coordinates of the extrema from S'(x)=0:
> `S'`(x):=Diff(FresnelS(xi),xi)=diff(FresnelS(x),x);
```

$$S'(x) := \frac{d}{d\xi}\,\mathrm{FresnelS}(\xi) = \sin\left(\frac{\pi x^2}{2}\right)$$

```
> plot({0,rhs(`S'`(x))},x=0..6,axes=boxed,color=black,
  title="Derivative S'(x)");
```




```
> # From S'(x)=sin(Pi*x^2/2)=0 we arrive at:
> X[extrem][i]:=sqrt(2*i);
```

$$X_{extrem_i} := \sqrt{2}\,\sqrt{i}$$

```
> for i from 0 to 20 do X[extrema][i]:=sqrt(2*i) od:
> with(LinearAlgebra):
> X[extrem]:=vector([seq(X[extrema][i],i=0..20)]);
```

$$X_{extrem} := [0, \sqrt{2}, 2, \sqrt{6}, 2\sqrt{2}, \sqrt{10}, 2\sqrt{3}, \sqrt{14}, 4, 3\sqrt{2}, 2\sqrt{5}, \sqrt{22}, 2\sqrt{6}, \sqrt{26}, 2\sqrt{7}, \sqrt{30}, 4\sqrt{2}, \sqrt{34}, 6, \sqrt{38}, 2\sqrt{10}]$$

```
> x[max]:=vector([seq(X[extrem][i],i=2..20,2)]);
  # x-coordinates of the maxima:
```

$$x_{max} := [\sqrt{2}, \sqrt{6}, \sqrt{10}, \sqrt{14}, 3\sqrt{2}, \sqrt{22}, \sqrt{26}, \sqrt{30}, \sqrt{34}, \sqrt{38}]$$

```
> x[max]:=evalf(%,4);
```

$$x_{max} := [1.414, 2.449, 3.162, 3.742, 4.242, 4.690, 5.099, 5.477, 5.831, 6.164]$$

```
> x[min]:=vector([seq(X[extrem][i],i=1..20,2)]);
  # x-coordinates of the minima:
```

$$x_{min} := [0, 2, 2\sqrt{2}, 2\sqrt{3}, 4, 2\sqrt{5}, 2\sqrt{6}, 2\sqrt{7}, 4\sqrt{2}, 6]$$

```
> x[min]:=evalf(%,4);
```

$$x_{min} := [0., 2., 2.828, 3.464, 4., 4.472, 4.898, 5.292, 5.656, 6.]$$

alternative:

```
> restart:  with(Statistics):
> x[max][i]:=sqrt(4*i-2);
```

$$x_{max_i} := \sqrt{4\,i - 2}$$

```
> X:=evalf(vector([seq(sqrt(4*i-2),i=1..10)]),4);
  # x-coordinates of the maxima:
```

$$X := [1.414, 2.449, 3.162, 3.742, 4.242, 4.690, 5.099, 5.477, 5.831, 6.164]$$

```
> whattype(X);
```

symbol

```
> # y-coordinates of the maxima:
> for i from 1 to 10 do
  S[max][i]:=evalf(subs(x=sqrt(4*i-2),FresnelS(x)),4) od:
> Y:=vector([seq(S[max][i],i=1..10)]);
```

$$Y := [0.7140, 0.6289, 0.6004, 0.5849, 0.5750, 0.5678, 0.5624, 0.5581, 0.5546, 0.5516]$$

```
> whattype(Y);
```

symbol

```
> DATA:=seq([X[i],Y[i]],i=1..10);
```

$$DATA := [1.414, 0.7140], [2.449, 0.6289], [3.162, 0.6004], [3.742, 0.5849], [4.242, 0.5750],$$
$$[4.690, 0.5678], [5.099, 0.5624], [5.477, 0.5581], [5.831, 0.5546], [6.164, 0.5516]$$

Nonlinear Regression: Again a "Four Parameters Double Exponential Decay" is a suitable nonlinear model function to DATA:

```
> f:=(t,A,B,C,D) -> 1/2+A*exp(-B*t)+C*exp(-D*t);
```

$$f := (t, A, B, C, \mathrm{D}) \to \frac{1}{2} + A\,\mathbf{e}^{(-B\,t)} + C\,\mathbf{e}^{(-\mathrm{D}\,t)}$$

```
> y[max]:=unapply(evalf(NonlinearFit(f(t,A,B,C,D),X,Y,t),4),t);
```

$$y_{max} := t \to 0.5000 + 0.1369\,\mathbf{e}^{(-0.1617\,t)} + 0.4375\,\mathbf{e}^{(-1.009\,t)}$$

```
> y[max](x):=subs(t=x,y[max](t));
```

$$y_{max}(x) := 0.5000 + 0.1369\,\mathbf{e}^{(-0.1617\,x)} + 0.4375\,\mathbf{e}^{(-1.009\,x)}$$

```
> y[max](0):=evalf(subs(x=0,%),4);
```

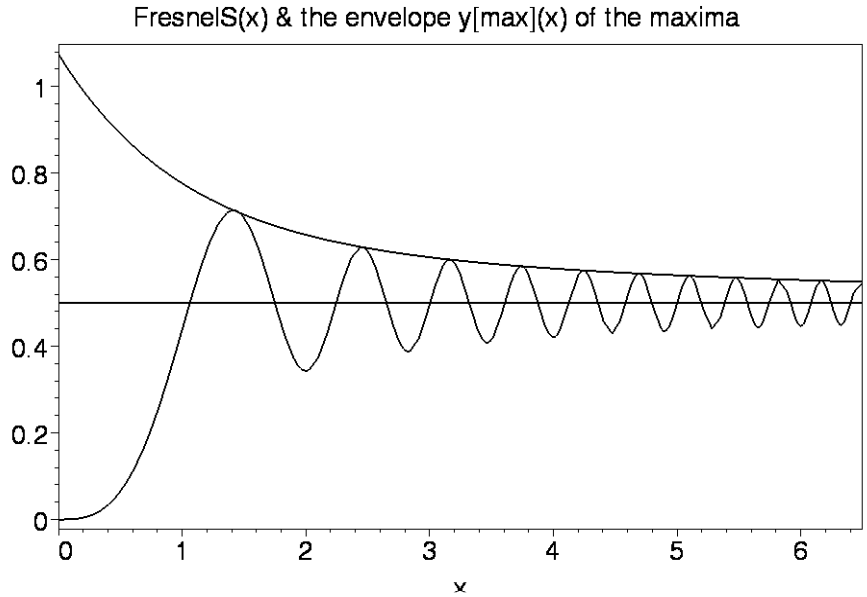
$$y_{max}(0) := 1.074$$

```
> y[max](infinity):=evalf(subs(x=infinity,%%),2);
```

$$y_{max}(\infty) := 0.50$$

```
> alias(th=thickness,co=color):
> plot({0.5,y[max](x),FresnelS(x)},x=0..6.5,axes=boxed,
```

```
th=2,co=black,
title="FresnelS(x) & the envelope y[max](x) of the maxima");
```



We see, the "Four Parameters Double Exponential Decay y[max](x)" is a suitable envelope of the maxima. However, an envelope of the minima cannot be expressed by a series of Exponential Decays because of the boundary condition in x = 0, which should be: y'[min](0) = 0. Thus, we propose a "Three Parameters Sigmoid" as a suitable envelope of the minima in the following:

```
> x[min][k]:=2*sqrt(k-1);
```

$$x_{min_k} := 2\sqrt{k-1}$$

```
> # x-coordinates of the minima added by x = 0.5:
> P:=evalf(array([0.5,seq(2*sqrt(i-1),i=1..10)]),4);
```

$$P := [0.5, 0., 2., 2.828, 3.464, 4., 4.472, 4.898, 5.292, 5.656, 6.]$$

```
> whattype(P);
```

symbol

```
> # additional value:
> FresnelS(0.5):=evalf(subs(x=0.5,FresnelS(x)),4);
```

$$\mathrm{FresnelS}(0.5) := 0.06473$$

```
> for i from 1 to 10 do
  S[min][i]:=evalf(subs(x=2*sqrt(i-1),FresnelS(x)),4) od:
> Q:=evalf(array([0.06473,seq(S[min][i],i=1..10)]));
  # y-coordinates of the minima:
```

$$Q := [0.06473, 0., 0.3434, 0.3880, 0.4083, 0.4205, 0.4289, 0.4351, 0.4399, 0.4438, 0.4470]$$

```
> whattype(Q);
```

symbol

```
> data:=seq([P[i],Q[i]],i=1..10);
```

$$data := [0.5, 0.06473], [0., 0.], [2., 0.3434], [2.828, 0.3880], [3.464, 0.4083], [4., 0.4205],$$
$$[4.472, 0.4289], [4.898, 0.4351], [5.292, 0.4399], [5.656, 0.4438]$$

For the lower envelope y[min](x) a "Three Parameters Sigmoid" is a suitable nonlinear model

function to the calculated data:

```
> y[min](t):=NonlinearFit(1/2-(1/2)/(1+(p*t)^q)^r,P,Q,t);
```

$$y_{min}(t) := \frac{1}{2} - \frac{1}{2\,(1 + 3.8767396996104\,t^{2.93806704062480817})^{0.338714430596105176}}$$

```
> y[min](x):=subs(t=x,y[min](t));
```

$$y_{min}(x) := \frac{1}{2} - \frac{1}{2\,(1 + 3.8767396996104\,x^{2.93806704062480817})^{0.338714430596105176}}$$

```
> y[min](0):=subs(x=0,y[min](x));
  y[min](infinity):=evalf(subs(x=infinity,y[min](x)),2);
```

$$y_{min}(0) := 0.$$

$$y_{min}(\infty) := 0.50$$

```
> `y'`[min](x):=diff(y[min](x),x);
```

$$y'_{min}(x) := \frac{1.928999198\,x^{1.938067041}}{(1 + 3.8767396996104\,x^{2.93806704062480817})^{1.338714431}}$$

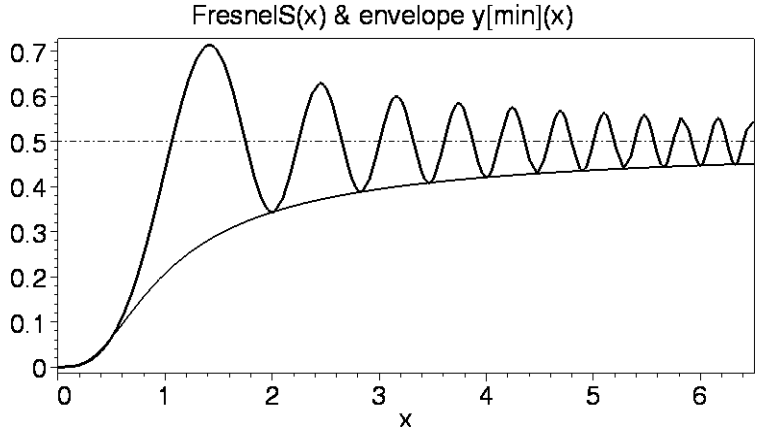
```
> `y'`[min](0):=subs(x=0,`y'`[min](x));
```

$$y'_{min}(0) := 0.$$

```
> `y'`[min](infinity):=evalf(subs(x=100000,`y'`[min](x)),4);
```

$$y'_{min}(\infty) := 0.3291\ 10^{-10}$$

```
> # The boundary conditions are fulfilled.
> alias(th=thickness,co=color):
> p[1]:=plot(1/2,x=0..6.5,linestyle=4,co=black,axes=boxed):
> p[2]:=plot(FresnelS(x),x=0..6.5,th=3,co=black):
> p[3]:=plot(y[min](x),x=0..6.5,th=2,co=black,
  title="FresnelS(x) & envelope y[min](x)"):
> plots[display](seq(p[k],k=1..3));
```



We see, the " Three Parameters Sigmoid" is a suitable envelope y[min](x) of the minima. In the following Figure the two envelopes have been represented.

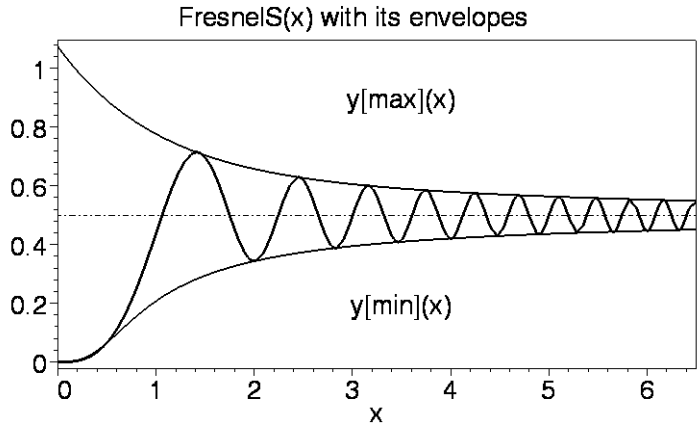
```
> y[max](x):=1/2+0.1369*exp(-0.1617*x)+0.4375*exp(-1.0089*x);
```

$$y_{max}(x) := \frac{1}{2} + 0.1369\ \mathbf{e}^{(-0.1617\,x)} + 0.4375\ \mathbf{e}^{(-1.0089\,x)}$$

```
> y[min](x):=1/2-1/(2*(1+3.8767*x^2.9381)^0.3387);
```

$$y_{min}(x) := \frac{1}{2} - \frac{1}{2\,(1 + 3.8767\,x^{2.9381})^{0.3387}}$$

```
> alias(th=thickness,co=color):
> p[1]:=plot(1/2,x=0..6.5,linestyle=4,co=black,axes=boxed):
> p[2]:=plot(FresnelS(x),x=0..6.5,th=3,co=black):
> p[3]:=plot({y[max](x),y[min](x)},x=0..6.5,th=2,co=black,
  title="FresnelS(x) with its envelopes"):
> p[4]:=plots[textplot]({[3.5,0.9,`y[max](x)`],
  [3.5,0.2,`y[min](x)`]}):
> plots[display](seq(p[k],k=1..4));
```



This Figure shows good approximations of the *envelopes* to both the maxima and the minima of *FresnelS(x)*, In the following the L[2] and l[2] error norms have been calculated for the approximations of the *nonlinear model functions*
y[max](x) & y[min](x) to DATA & data, respectively. First, we introduce the *distance vector* between the envelope
y[max](x) and the DATA of the maxima.

```
> with(linalg):
> for i from 1 to 10 do
  v[i]:=evalf(subs(x=DATA[i][1],y[max](x))-DATA[i][2]) od:

Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> V:=vector([seq(v[i],i=1..10)]);
```

$V := [-0.0000248844, 0.00021073979, -0.00028843883, -0.00011656702, 0.3362354\ 10^{-5},$
$0.000182648846, 0.000175413141, 0.000107453720, -0.000057269331, -0.0002004951869]$

```
> L[2]:=(1/sqrt(number_of_points))*Norm(V,2)=
  evalf((1/sqrt(10))*norm(V,2),4);
```

$$L_2 := \frac{\text{Norm}(V, 2)}{\sqrt{number\_of\_points}} = 0.0001615$$

```
> L[infinity]:=
  (1/(number_of_points)^(1/infinity))*Norm(V,infinity)=
  evalf((1/(number_of_points)^(1/infinity))*norm(V,infinity),4);
```

$$L_\infty := \text{Norm}(V, \infty) = 0.0002884$$

```
> L[infinity]:=Max(abs(v[1..10]))=evalf(abs(v[3]),4);
```

$$L_\infty := \text{Max}(|v_{1\,..\,10}|) = 0.0002884$$

```
> #  By the same way we arrive at l[2] error norm indicating the
  approximation of the nonlinear envelope y[min](x) to the
  coordinates of the minima [data].
> for i from 1 to 10 do
  w[i]:=evalf(subs(x=data[i][1],y[min](x))-data[i][2]) od:
> W:=vector([seq(w[i],i=1..10)]);
```

$$W := [0.16044\ 10^{-5}, 0., -0.0001556352, 0.0001600953, 0.0001364202, 0.0000886398,$$
$$0.21679\ 10^{-5}, -0.0000591802, -0.0000554879, -0.0001088364]$$

```
> l[2]:=(1/sqrt(number_of_points))*Norm(W,2)=
  evalf((1/sqrt(10))*norm(W,2),4);
```

$$l_2 := \frac{\text{Norm}(W, 2)}{\sqrt{number\_of\_points}} = 0.00009733$$

```
> l[infinity]:=
  (1/(number_of_points)^(1/infinity))*Norm(W,infinity)=
  evalf((1/(number_of_points)^(1/infinity))*norm(W,infinity),4);
```

$$l_\infty := \text{Norm}(W, \infty) = 0.0001601$$

```
> l[infinity]:=Max(abs(w[1..10]))=evalf(abs(w[4]),4);
```

$$l_\infty := \text{Max}(|w_{1\,..\,10}|) = 0.0001601$$

```
> # general:
> L[p]:=(1/(number_of_points)^(1/p))*Norm(V,p);
```

$$L_p := \frac{\text{Norm}(V, p)}{number\_of\_points^{\left(\frac{1}{p}\right)}}$$

```
> l[p]:=(L[p]/Norm(V,p))*Norm(W,p);
```

$$l_p := \frac{\text{Norm}(W, p)}{number\_of\_points^{\left(\frac{1}{p}\right)}}$$

The *error norms* show that we have determined suitable *nonlinear model functions* for the *envelopes* y[max](x) and y[min](x) of *Fresnel's integral S(x)* represented in the obove Figure. In the same way one can calculate error norms for the envelopes of *Fresnel's integral C(x).*

The next example is concerned with the ***Harding of Aluminium Alloy***
Some tensile tests on *Aluminium Alloy AA 7075 T 7351* at room teperature furnish the following experimentl data.

```
> restart:  with(Statistics):
> data:=[0,350],[0.32,472],[1.27,501.2],[2.23,517],[3.2,530],[4.17
  ,541];
```

$$data := [0, 350], [0.32, 472], [1.27, 501.2], [2.23, 517], [3.2, 530], [4.17, 541]$$

This hardening behaviour can be expressed by the simple formula:

```
> y(xi):=a+b*xi^c;
```

$$y(\xi) := a + b\,\xi^{c}$$

```
> X:=vector([seq(data[i][1],i=1..6)]);
```

$$X := [0, 0.32, 1.27, 2.23, 3.2, 4.17]$$

```
> Y:=vector([seq(data[i][2],i=1..6)]);
```

$$Y := [350, 472, 501.2, 517, 530, 541]$$

```
> y(t):=evalf(NonlinearFit(350+b*t^c,X,Y,t),4);
```

$$y(t) := 350. + 146.8\,t^{0.175786888339528214}$$

```
> y(x):=350+146.8*x^0.1758;
```

$$y(x) := 350 + 146.8\,x^{0.1758}$$

The parameter [a, b, c] can be interpreted as:

```
> a:=yield_stress=350*MPa; b:=hardening_factor=146.8;
  c:=hardening_exponent=0.1758;
```

$$a := yield\_stress = 350\ MPa$$

$$b := hardening\_factor = 146.8$$

$$c := hardening\_exponent = 0.1758$$

Dimensionless representation in both coordinates (x & y):

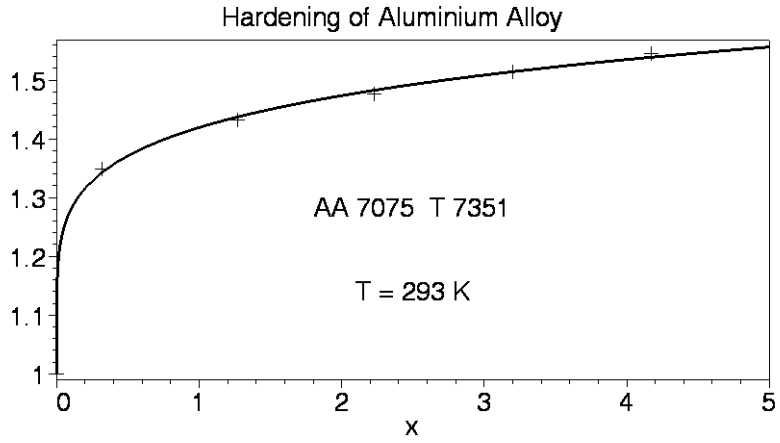
```
> z(x):=y(y)/yield_stress=evalf(y(x)/350,4);
```

$$z(x) := \frac{y(y)}{yield\_stress} = 1. + 0.4194\,x^{0.1758}$$

```
> DATA:=[0,1],[0.32,472./350],[1.27,501.2/350],[2.23,517./350],
  [3.2,530./350],[4.17,541./350];
```

$$DATA := [0, 1], [0.32, 1.348571429], [1.27, 1.432000000], [2.23, 1.477142857], [3.2, 1.514285714], [4.17, 1.545714286]$$

```
> alias(th=thickness,co=color):
> p[1]:=plot(rhs(z(x)),x=0..5,th=3,co=black,axes=boxed,
  title="Hardening of Aluminium Alloy"):
> p[2]:=plot([DATA],x=0..5,th=3,co=black,
  style=point,symbol=cross,symbolsize=30):
> p[3]:=plots[textplot]({[2.5,45/35,`AA 7075  T 7351`],
  [2.5,40/35,`T = 293 K`]}):
> plots[display](seq(p[k],k=1..3));
```



This Figure shows a good approximation to the experimental data. Furthermore, let us calculate the L[2] vector norm expressing the *distance* between the nonlinear model function & the experimental data.

```
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> for i from 1 to 6 do
  w[i]:=evalf(subs(x=DATA[i][1],rhs(z(x))-DATA[i][2])) od:
> W:=vector([seq(w[i],i=1..6)]);
```

$$W := [0., -0.0053030554, 0.0053983296, 0.0057607186, 0.0002716433, -0.0066403372]$$

```
> L[2]:=(1/sqrt(number_of_points))*Norm(W,2)=
  evalf((1/sqrt(6))*norm(W,2),4);
```

$$L_2 := \frac{\mathrm{Norm}(W, 2)}{\sqrt{number\_of\_points}} = 0.004734$$

Thus, the above approximation of the model function to the hardening DATA is very suitable. In the next worksheet a nonlinear polynomial has been assumed as a model function fitting to a given data set.

```
> restart:
> with(Statistics):
> data:=[0,20],[1,30],[2,27],[3,25],[4,24],[5,21],[6,19];
```

$$data := [0, 20], [1, 30], [2, 27], [3, 25], [4, 24], [5, 21], [6, 19]$$

```
> # polynomial:
> y(xi):=sum(a[k]*xi^k,k=0..6);
```

$$y(\xi) := a_0 + a_1\,\xi + a_2\,\xi^2 + a_3\,\xi^3 + a_4\,\xi^4 + a_5\,\xi^5 + a_6\,\xi^6$$

```
> X:=vector([seq(data[i][1],i=1..7)]);
```

$$X := [0, 1, 2, 3, 4, 5, 6]$$

```
> Y:=vector([seq(data[i][2],i=1..7)]);
```

$$Y := [20, 30, 27, 25, 24, 21, 19]$$

```
> y(t):=evalf(NonlinearFit(sum(a[k]*t^k,k=0..6),X,Y,t));
```

$$y(t) := 20. + 27.20000000\,t - 25.26111111\,t^2 + 9.666666667\,t^3 - 1.736111111\,t^4$$

$$+ 0.1333333333\,t^5 - 0.002777777778\,t^6$$

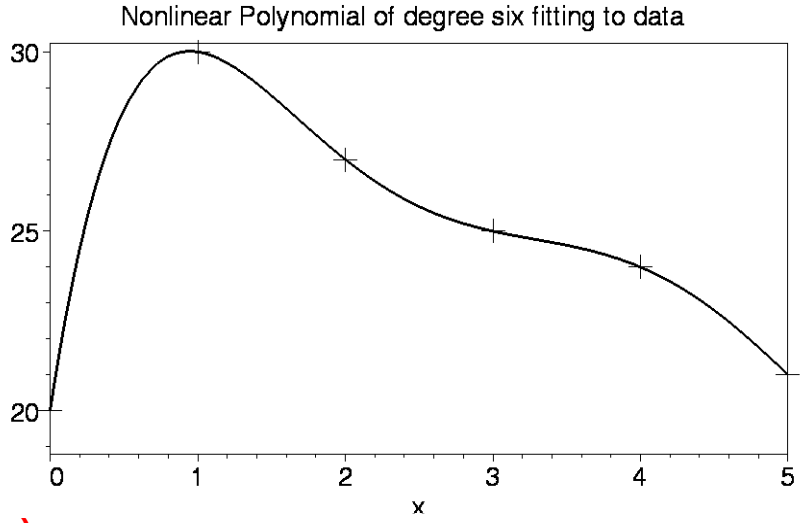
```
> z(t):=convert(y(t),rational);
```

$$z(t) := 20 + \frac{136}{5}t - \frac{4547}{180}t^2 + \frac{29}{3}t^3 - \frac{125}{72}t^4 + \frac{2}{15}t^5 - \frac{1}{360}t^6$$

```
> y(x):=subs(t=x,z(t));
```

$$y(x) := 20 + \frac{136}{5}x - \frac{4547}{180}x^2 + \frac{29}{3}x^3 - \frac{125}{72}x^4 + \frac{2}{15}x^5 - \frac{1}{360}x^6$$

```
> alias(th=thickness,co=color):
> p[1]:=plot(y(x),x=0..5,th=3,co=black,axes=boxed,
  title="Nonlinear Polynomial of degree six fitting to data"):
> p[2]:=plot([data],style=point,symbol=cross,symbolsize=50,
  th=3,co=black,ytickmarks=3):
> plots[display](seq(p[k],k=1..2));
```




```
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
> for i from 1 to 7 do
  v[i]:=subs(x=data[i][1],y(x))-data[i][2] od:
> V:=vector([seq(v[i],i=1..7)]);
```

$$V := [0, 0, 0, 0, 0, 0, 0]$$

```
> L[2]:=(1/sqrt(number_of_points))*Norm(V,2)=(1/sqrt(7))*norm(V,2)
  ;
```

$$L_2 := \frac{\mathrm{Norm}(V, 2)}{\sqrt{number\_of\_points}} = 0$$

The *error norm* shows that the model function y(x) is very suitable fitting to the given data.

Alternatively to the Statistics [NonlinearFit] one can find an interpolation polynomial suitable for fitting to the given data by using the following *determinant-method.*

```
> restart:
> with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> M:=
  matrix(8,8,[[P[6](x),seq(x^i,i=0..6)],
  [y[0],seq(x[0]^i,i=0..6)], [y[1],seq(x[1]^i,i=0..6)],
  [y[2],seq(x[2]^i,i=0..6)], [y[3],seq(x[3]^i,i=0..6)],
  [y[4],seq(x[4]^i,i=0..6)], [y[5],seq(x[5]^i,i=0..6)],
  [y[6],seq(x[6]^i,i=0..6)]]);
```

$$M := \begin{bmatrix} P_6(x) & 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 \\ y_0 & 1 & x_0 & x_0^2 & x_0^3 & x_0^4 & x_0^5 & x_0^6 \\ y_1 & 1 & x_1 & x_1^2 & x_1^3 & x_1^4 & x_1^5 & x_1^6 \\ y_2 & 1 & x_2 & x_2^2 & x_2^3 & x_2^4 & x_2^5 & x_2^6 \\ y_3 & 1 & x_3 & x_3^2 & x_3^3 & x_3^4 & x_3^5 & x_3^6 \\ y_4 & 1 & x_4 & x_4^2 & x_4^3 & x_4^4 & x_4^5 & x_4^6 \\ y_5 & 1 & x_5 & x_5^2 & x_5^3 & x_5^4 & x_5^5 & x_5^6 \\ y_6 & 1 & x_6 & x_6^2 & x_6^3 & x_6^4 & x_6^5 & x_6^6 \end{bmatrix}$$

```
> data:=[0,20],[1,30],[2,27],[3,25],[4,24],[5,21],[6,19];
```

$$data := [0, 20], [1, 30], [2, 27], [3, 25], [4, 24], [5, 21], [6, 19]$$

```
> m:=matrix(8,8,[[P[6](x),seq(x^k,k=0..6)],[20,seq(0^k,k=0..6)],
  [30,seq(1^k,k=0..6)],[27,seq(2^k,k=0..6)],[25,seq(3^k,k=0..6)],
  [24,seq(4^k,k=0..6)],[21,seq(5^k,k=0..6)],[19,seq(6^k,k=0..6)]])
  ;
```

$$m := \begin{bmatrix} P_6(x) & 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 \\ 20 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 30 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 27 & 1 & 2 & 4 & 8 & 16 & 32 & 64 \\ 25 & 1 & 3 & 9 & 27 & 81 & 243 & 729 \\ 24 & 1 & 4 & 16 & 64 & 256 & 1024 & 4096 \\ 21 & 1 & 5 & 25 & 125 & 625 & 3125 & 15625 \\ 19 & 1 & 6 & 36 & 216 & 1296 & 7776 & 46656 \end{bmatrix}$$

```
> Polynom_P[6](xi):=Solve(Det(M)=0,P[6](xi));
```

$$Polynom\_P_6(\xi) := \mathrm{Solve}(\mathrm{Det}(M) = 0, P_6(\xi))$$

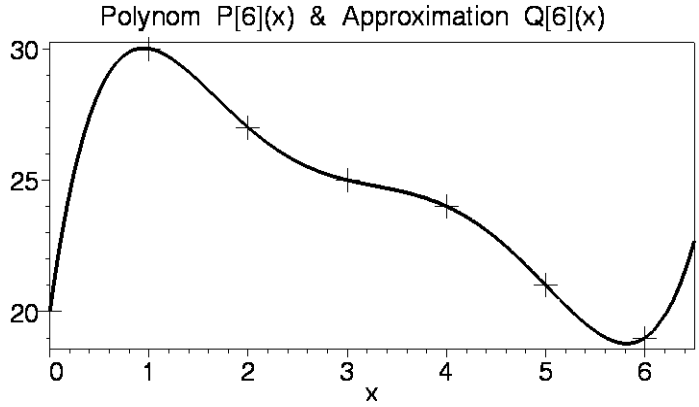
```
> Polynom[6]:=solve(det(m)=0,P[6](x)):
> P[6](x):= 20+(136/5)*x-(4547/180)*x^2+(29/3)*x^3-(125/72)*x^4+
  (2/15)*x^5-(1/360)*x^6;
```

$$P_6(x) := 20 + \frac{136}{5}x - \frac{4547}{180}x^2 + \frac{29}{3}x^3 - \frac{125}{72}x^4 + \frac{2}{15}x^5 - \frac{1}{360}x^6$$

```
> #  approximation with respect to Digits = 4:
> Q[6](x):=evalf(P[6](x),4);
```

$$Q_6(x) := 20. + 27.20\,x - 25.26\,x^2 + 9.667\,x^3 - 1.736\,x^4 + 0.1333\,x^5 - 0.002778\,x^6$$

```
> alias(th=thickness,co=color):
> p[1]:=plot(P[6](x),x=0..6.5,th=4,co=black,axes=boxed,ytickmarks=
  3):
> p[2]:=plot(Q[6](x),x=0..6.5,linestyle=3,th=3,co=black):
> p[3]:=plot([data],x=0..6.5,style=point,symbol=cross,
  symbolsize=50,th=3,co=black,
  title="Polynom  P[6](x)  &  Approximation  Q[6](x)" ):
> plots[display](seq(p[k],k=1..3));
```



In this Figure we cannot see the distance between the Polynomial P[6](x) and the Approximation Q[6](x). Therefore, we calculate the distance:

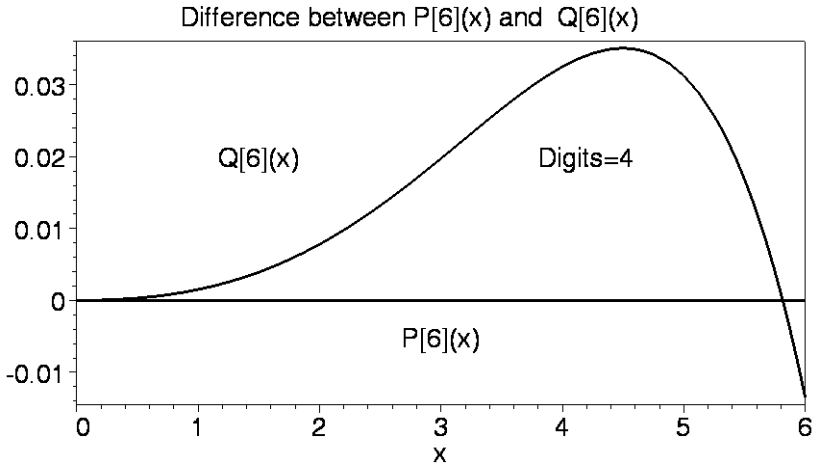
```
> for i from 1 to 7 do
  v[i]:=subs(x=data[i][1],P[6](x))-data[i][2] od:
> V:=vector([seq(v[i],i=1..7)]);
```

$$V := [0, 0, 0, 0, 0, 0, 0]$$

```
> # Thus, the Polynomial P[6](x) is exact. The difference between
  P[6](x) und the approximation Q[6](x) is given as follows:
> Delta(x):=Q[6](x)-P[6](x);
```

$$\Delta(x) := 0.00111111\,x^2 + 0.000333333\,x^3 + 0.000111111\,x^4 - 0.0000333333\,x^5 - 0.222222\,10^{-6}\,x^6$$

```
> alias(th=thickness,co=color):
> p[1]:=plot({0,Delta(x)},x=0..6,th=3,co=black,axes=boxed,
  title="Difference between P[6](x) and  Q[6](x)"):
> p[2]:=plots[textplot]({[1.5,0.02,`Q[6](x)`],
  [4.2,0.02,`Digits=4`],[3,-0.005,`P[6](x)`]}):
> plots[display](seq(p[k],k=1..2));
```



This Figure shows that the maximal error of the approximation with Digits=4 is more than 3 % .
Another method generating *interpolating polynomials* is due to ***LAGRANGE*** as follows:

```
> restart:
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
> x[k]:=matrix(1,7,[seq(k,k=0..6)]);
```

$$x_k := [0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6]$$

```
> y[k]:=matrix(1,7,[20,30,27,25,24,21,19]);
```

$$y_k := [20 \quad 30 \quad 27 \quad 25 \quad 24 \quad 21 \quad 19]$$

```
> data:=[0,20],[1,30],[2,27],[3,25],[4,24],[5,21],[6,19];
```

$$data := [0, 20], [1, 30], [2, 27], [3, 25], [4, 24], [5, 21], [6, 19]$$

```
> #  Lagrange interpolating polynomial of degree six:
> L[6,k](x):=Product((x-x[i])/(x[k]-x[i]),i=0..6);  k<>i;
```

$$L_{6,k}(x) := \prod_{i=0}^{6} \frac{x - x_i}{x_k - x_i}$$

$$k \neq i$$

```
> L[6,k](x):=value(L[6,k](x));
```

$$L_{6,k}(x) := \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)(x-x_5)(x-x_6)}{(x_k-x_0)(x_k-x_1)(x_k-x_2)(x_k-x_3)(x_k-x_4)(x_k-x_5)(x_k-x_6)}$$

```
> for i from 0 to 6 do L[6,i](x):=L[6,k](x)*(x[k]-x[i])/(x-x[i])
  od:
> for i from 0 to 6 do L[6,i](x):=
  expand(subs({x[k]=x[i]},{seq(x[m]=m,m=0..6)},L[6,i](x))) od;
```

$$L_{6,0}(x) := \frac{1}{720}x^6 - \frac{7}{240}x^5 + \frac{35}{144}x^4 - \frac{49}{48}x^3 + \frac{203}{90}x^2 - \frac{49}{20}x + 1$$

$$L_{6,1}(x) := -\frac{1}{120}x^6 + \frac{1}{6}x^5 - \frac{31}{24}x^4 + \frac{29}{6}x^3 - \frac{87}{10}x^2 + 6x$$

$$L_{6,2}(x) := \frac{1}{48}x^6 - \frac{19}{48}x^5 + \frac{137}{48}x^4 - \frac{461}{48}x^3 + \frac{117}{8}x^2 - \frac{15}{2}x$$

$$L_{6,3}(x) := -\frac{1}{36}x^6 + \frac{1}{2}x^5 - \frac{121}{36}x^4 + \frac{31}{3}x^3 - \frac{127}{9}x^2 + \frac{20}{3}x$$

$$L_{6,4}(x) := \frac{1}{48}x^6 - \frac{17}{48}x^5 + \frac{107}{48}x^4 - \frac{307}{48}x^3 + \frac{33}{4}x^2 - \frac{15}{4}x$$

$$L_{6,5}(x) := -\frac{1}{120}x^6 + \frac{2}{15}x^5 - \frac{19}{24}x^4 + \frac{13}{6}x^3 - \frac{27}{10}x^2 + \frac{6}{5}x$$

$$L_{6,6}(x) := \frac{1}{720}x^6 - \frac{1}{48}x^5 + \frac{17}{144}x^4 - \frac{5}{16}x^3 + \frac{137}{360}x^2 - \frac{1}{6}x$$

```
> #  Polynom of degree six:
> P[6](x):=Sum(y[kappa]*L[6,kappa](x),kappa=0..6);
```

$$P_6(x) := \sum_{\kappa=0}^{6} y_\kappa L_{6,\kappa}(x)$$

```
> L[6,k](x):=matrix(7,1,[seq(L[6,k](x),k=0..6)]);
```

$$L_{6,k}(x) := \begin{bmatrix} \frac{1}{720}x^6 - \frac{7}{240}x^5 + \frac{35}{144}x^4 - \frac{49}{48}x^3 + \frac{203}{90}x^2 - \frac{49}{20}x + 1 \\ -\frac{1}{120}x^6 + \frac{1}{6}x^5 - \frac{31}{24}x^4 + \frac{29}{6}x^3 - \frac{87}{10}x^2 + 6x \\ \frac{1}{48}x^6 - \frac{19}{48}x^5 + \frac{137}{48}x^4 - \frac{461}{48}x^3 + \frac{117}{8}x^2 - \frac{15}{2}x \\ -\frac{1}{36}x^6 + \frac{1}{2}x^5 - \frac{121}{36}x^4 + \frac{31}{3}x^3 - \frac{127}{9}x^2 + \frac{20}{3}x \\ \frac{1}{48}x^6 - \frac{17}{48}x^5 + \frac{107}{48}x^4 - \frac{307}{48}x^3 + \frac{33}{4}x^2 - \frac{15}{4}x \\ -\frac{1}{120}x^6 + \frac{2}{15}x^5 - \frac{19}{24}x^4 + \frac{13}{6}x^3 - \frac{27}{10}x^2 + \frac{6}{5}x \\ \frac{1}{720}x^6 - \frac{1}{48}x^5 + \frac{17}{144}x^4 - \frac{5}{16}x^3 + \frac{137}{360}x^2 - \frac{1}{6}x \end{bmatrix}$$

```
> P[6](x):=multiply(y[k],L[6,k](x));
```

$$P_6(x) := \left[20 + \frac{136}{5}x - \frac{4547}{180}x^2 + \frac{29}{3}x^3 + \frac{2}{15}x^5 - \frac{1}{360}x^6 - \frac{125}{72}x^4\right]$$

```
> P[6](x):=20+(136/5)*x-(4547/180)*x^2+(29/3)*x^3-
  (125/72)*x^4+(2/15)*x^5-(1/360)*x^6;
```

$$P_6(x) := 20 + \frac{136}{5}x - \frac{4547}{180}x^2 + \frac{29}{3}x^3 + \frac{2}{15}x^5 - \frac{1}{360}x^6 - \frac{125}{72}x^4$$

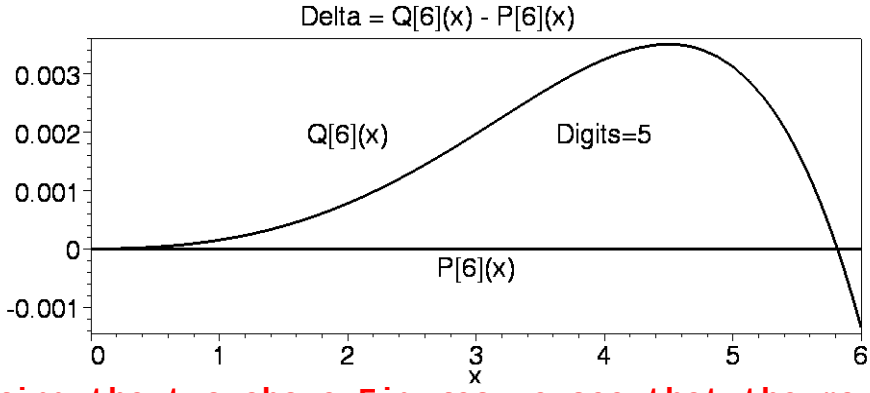
```
> # This polynom is identical to the above result.
> # Approximation by Digits=5:
> Q[6][5](x):=evalf(P[6](x),5);
```

$$Q_{6_5}(x) := 20. + 27.200\,x - 25.261\,x^2 + 9.6667\,x^3 + 0.13333\,x^5 - 0.0027778\,x^6 - 1.7361\,x^4$$

```
> Delta(x):=Q[6][5](x)-P[6](x);
```

$$\Delta(x) := 0.00011111\,x^2 + 0.000033333\,x^3 - 0.33333\,10^{-5}\,x^5 - 0.22222\,10^{-7}\,x^6 + 0.000011111\,x^4$$

```
> alias(th=thickness,co=color):
> p[1]:=plot({0,Delta(x)},x=0..6,ytickmarks=4,
  axes=boxed,th=3,co=black,
  title="Delta = Q[6](x) - P[6](x)"):
> p[2]:=plots[textplot]({[3,-0.0003,`P[6](x)`],
  [2,0.002,`Q[6](x)`],[4,0.002,`Digits=5`]}):
> plots[display](seq(p[k],k=1..2));
```




```
> #  Comparing the two above Figures we see that the maximal
  errors of the approximations with Digits=4 and Digits=5 are more
  than 0.03 and 0.003, respectively.
```

Summary

This paper is concerned with applications of *Nonlinear Regressions.* Based upon given sets of data some examples have been applied, e.g., the envelopes of *Fresnel's integrals* *C(x)* and *S(x)* or the *hardening of aluminium alloy* ,etc. *Numerical effects* of approximations have also discussed by assuming several values of the Maple variable ***Digits*** in ***evalf*** (evaluate using floating-point arithmethic), whose default value is equal to ten.

In the examples discussed in this paper the following *nonlinear model functions* fit to data have been introduced:

--- Four-Parameter Double Exponential Decay,
--- Three-Parameter Sigmoid,
--- Non Linear Interpolating Polynomials of Degree Six due to *Lagrange* or using the Determinant Method.

The calculated *error norms* show that we have determined suitable *non linear model functions* fit to given data sets.


```
>
```