```
>
```

February 2016

Global Population from 1804 to 2015

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

This worsheet is concerned with the development of the global population during the period of 1804 - 2015. In 1804 or 2015 the population is 10^9 or 7.4*10^9, respectively. The given data has been interpolated by the *cubic spline function.*
Several nonlinear model functions to data have been suggested and tested by using *error norms.* The arguments in the approximation functions are given by:


```
> restart:
> tau(xi):=(xi-1804)/211; tau(1804):=0; tau(2015):=7.4;
```

$$\tau(\xi) := \frac{\xi}{211} - \frac{1804}{211}$$

$$\tau(1804) := 0$$

$$\tau(2015) := 7.4$$

Cubic Spline Interpolation

```
> restart:
> with(Statistics):
> DATA:=   [1804,1],[1922,2],[1955,3],[1974,4],[1987,5],
  [1999,6],[2010,7],[2015,7.4];
```

$$DATA := [1804, 1], [1922, 2], [1955, 3], [1974, 4], [1987, 5], [1999, 6], [2010, 7], [2015, 7.4]$$

```
> X:=array([1804,1922,1955,1974,1987,1999,2010,2015]);
```

$$X := [1804, 1922, 1955, 1974, 1987, 1999, 2010, 2015]$$

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

$$symbol$$

```
> Y:=array([seq(i,i=1..7),7.4]);
```

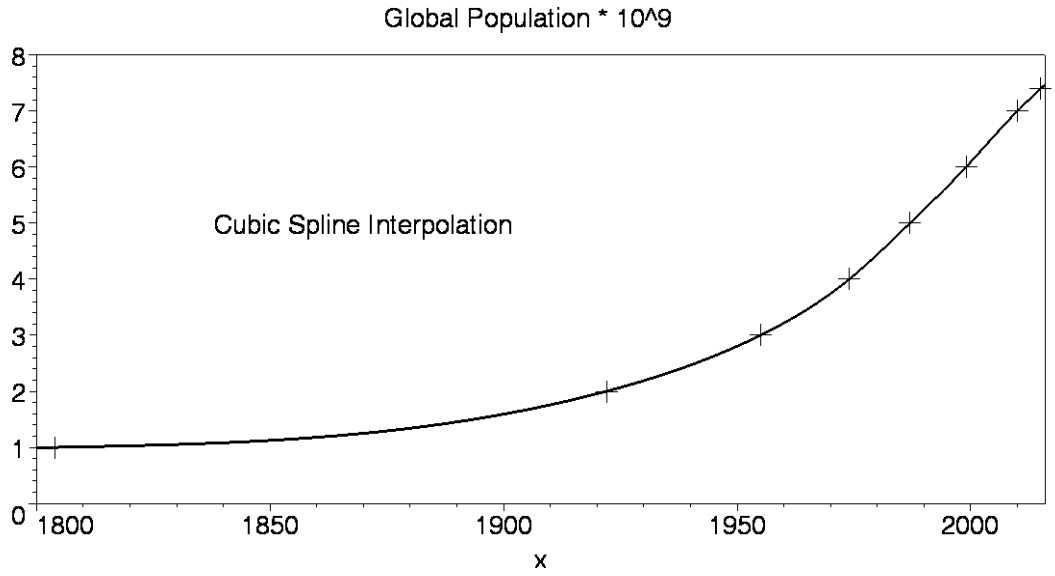
$$Y := [1, 2, 3, 4, 5, 6, 7, 7.4]$$

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

$$symbol$$

```
> with(CurveFitting):
```

```
> Sp(x):=Spline([DATA],x,degree=3):
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1800..2016,0..8,th=3,co=black,
  style=point,symbol=cross,symbolsize=45,axes=boxed,
  title="Global Population * 10^9"):
> p[2]:=plot(Sp(x),x=1800..2016,th=3,co=black):
> p[3]:=plots[textplot]([1870,5,`Cubic Spline Interpolation`]):
> plots[display](seq(p[k],k=1..3));
```



Approximation y(x): Three Parameter Single Exponential Groth

```
> f:=(t,a,b,c) -> a + b*exp(c*(t-1804)/211);
```

$$f := (t, a, b, c) \rightarrow a + b\, \mathbf{e}^{(1/211\, c\, (t - 1804))}$$

```
> y:=unapply(evalf(NonlinearFit(f(t,a,b,c),X,Y,t)),t);
```

$$y := t \rightarrow 0.780667500337245 + 0.157499785429422\, \mathbf{e}^{(0.0178166440516068\, t - 32.1412258690987)}$$

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

$$y(x) := 0.780667500337245 + 0.157499785429422\, \mathbf{e}^{(0.0178166440516068\, x - 32.1412258690987)}$$

```
> y(1804):=evalf(subs(x=1804,y(x)));
```

$$y(1804) := 0.938167285766667$$

```
> y(1804)[data]:=1;
```

$$y(1804)_{data} := 1$$

```
> y(2015):=evalf(subs(x=2015,y(x)));
```

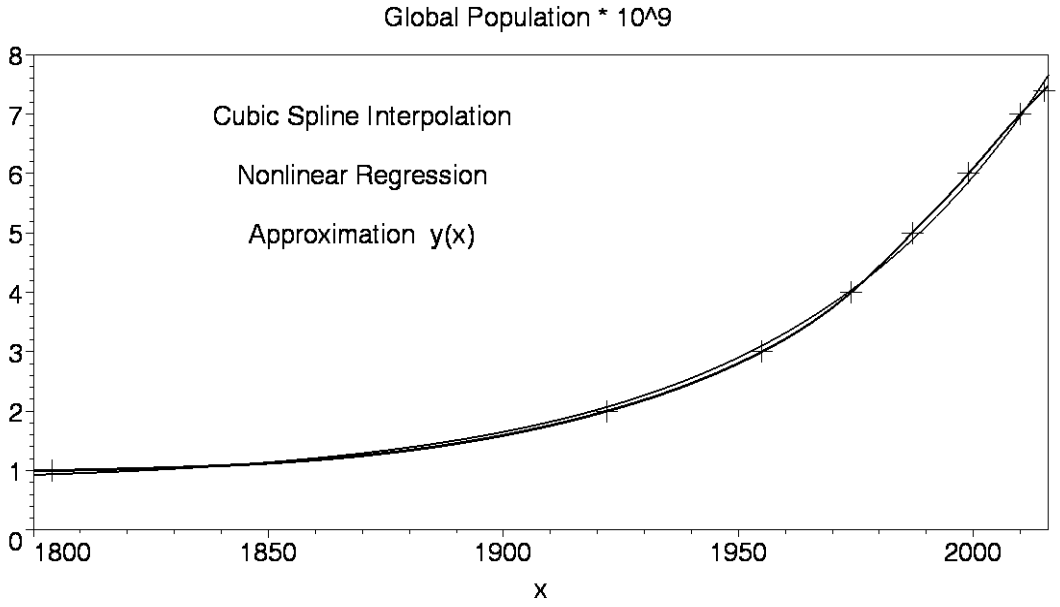
$$y(2015) := 7.54038238103267$$

```
> y(2015)[data]:=7.4;
```

$$y(2015)_{data} := 7.4$$

```
>
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1800..2016,0..8,axes=boxed,
  th=3,co=black,style=point,symbol=cross,symbolsize=45):
```

```
> p[2]:=plot(Sp(x),x=1800..2016,th=3,co=black,
  title="Global Population * 10^9"):
> p[3]:=plot(y(x),x=1800..2016,th=2,co=black):
> p[4]:=plots[textplot]([[1870,7,`Cubic Spline Interpolation`],
  [1870,6,`Nonlinear Regression`],[1870,5,`Approximation
  y(x)`]]):
> plots[display](seq(p[k],k=1..4));
```



The L[2] *error norm* between the cubic spline interpolation and the approximation y(x) can be expressed as:

```
> L[2]:=
  sqrt((1/211)*Int((SPLINE-ypsilon(x))^2,x=1804..2015))=
  evalf(sqrt((1/211)*int((Sp(x)-y(x))^2,x=1804..2015)));
```

$$L_2 := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} (SPLINE - \text{ypsilon}(x))^2\, dx} = 0.07131679787$$

The l[2] *error norm* of the approximation y(x) with respect to the data is given by:

```
> with(linalg):
> for i from 1 to 8 do
  v[i]:=evalf(subs(x=DATA[i][1],y(x))-DATA[i][2]) od:
> V:=vector([seq(v[i],i=1..8)]);
```

$V := [-0.0618327142333333, 0.0698816095127985, 0.101630229137681,$
$0.0366637806652945, -0.114706301969693, -0.136267597200296, -0.0357518828930905,$
$0.140382381032667]$

```
> l[2]:=
  (1/sqrt(number_of_points))*Norm(V,2)=
  evalf((1/sqrt(8))*norm(V,2));
```

$$l_2 := \frac{\mathrm{Norm}(V, 2)}{\sqrt{number\_of\_points}} = 0.0955846901570484$$

The above approximation y(x) could be improved by introducing the model function z(x) , where the boundary conditions z(1804) = 1 and z(2015) = 7.4 are apriori fulfilled exactly.

```
> z(x):= 1-(6.4/(exp(d)-1))*(1-exp(d*(x-1804)/211));
```

$$z(x) := 1 - \frac{6.4\left(1 - \mathbf{e}^{\left(\frac{d\,(x-1804)}{211}\right)}\right)}{\mathbf{e}^{d} - 1}$$

```
> z(x):=evalf(NonlinearFit(z(x),X,Y,x));
```

$$z(x) := 0.835636845976257 + 0.164363154023743\ \mathbf{e}^{(0.0174755092048838\ x\ -\ 31.5258186056104)}$$

```
> z(1804):=simplify(subs(x=1804,z(x)));
```

$$z(1804) := 1.$$

```
> z(1804)[data]:=1;
```

$$z(1804)_{data} := 1$$

```
> z(2015):=simplify(subs(x=2015,z(x)));
```

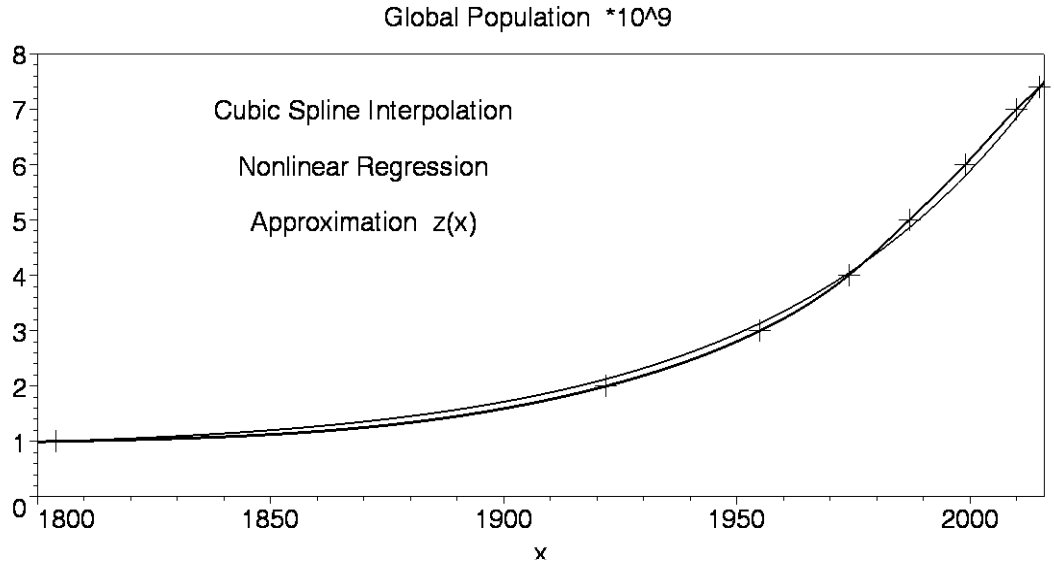
$$z(2015) := 7.399999999$$

```
> z(2015)[data]:=7.4;
```

$$z(2015)_{data} := 7.4$$

Note: With Digits = default we arrive at the exact boundary conditions.

```
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1800..2016,0..8,axes=boxed,
  th=3,co=black,style=point,symbol=cross,symbolsize=45):
> p[2]:=plot(Sp(x),x=1800..2016,th=3,co=black,
  title="Global Population  *10^9"):
> p[3]:=plot(z(x),x=1800..2016,th=2,co=black):
> p[4]:=plots[textplot]({[1870,7,`Cubic Spline Interpolation`],
  [1870,6,`Nonlinear Regression`],
  [1870,5,`Approximation  z(x)`]}):
> plots[display](seq(p[k],k=1..4));
```



The M[2] *error norm* between the spline interpolation and the approximation z(x) can be expressed as:

```
> M[2]:=
  sqrt((1/211)*Int((SPLINE -Z(xi))^2,xi=1804..2015))=
  evalf(sqrt((1/211)*int((Sp(x)-z(x))^2,x=1804..2015)));
```

$$M_2 := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} (SPLINE - Z(\xi))^2\, d\xi} = 0.1149950158$$

The m[2] *error norm* of the approximation z(x) with respect to the data is given by:

```
> with(linalg):
> for i from 1 to 8 do
  w[i]:=evalf(subs(x=DATA[i][1],z(x))-DATA[i][2]) od:
> W:=vector([seq(w[i],i=1..8)]);
```

W := [0., 0.127949205857758, 0.136133288978813, 0.0420714970464786,

-0.140102365694109, -0.201179753302434, -0.149233220736058, $-0.806045896695196\ 10^{-9}$]

```
> m[2]:=
  (1/sqrt(number_of_points))*Norm(W,2)=
  evalf((1/sqrt(8))*norm(W,2));
```

$$m_2 := \frac{\mathrm{Norm}(W, 2)}{\sqrt{number\_of\_points}} = 0.121986578478900$$

Exponential Groth Multiplied with [(t-1804)/211]^2 is the next Example.

```
> h:=(t,a)->1+(6.4/exp(a))*(((t-1804)/211)^2)*exp(a*(t-1804)/211);
```

$$h := (t, a) \to 1 + \frac{6.4\left(\frac{1}{211}t - \frac{1804}{211}\right)^2 \mathbf{e}^{(1/211\, a\, (t-1804))}}{\mathbf{e}^a}$$

```
> r:=unapply(evalf(NonlinearFit(h(t,a),X,Y,t)),t);
```

$$r := t \rightarrow 1. + 1.3540749879374\,(0.004739336493\,t - 8.549763033)^2\,\mathbf{e}^{(0.00736103997575911\,t - 13.2793161162694)}$$

```
> r(x):=subs(t=x,r(t));
```

$$r(x) := 1. + 1.35407498793740\,(0.004739336493\,x - 8.549763033)^2\,\mathbf{e}^{(0.00736103997575911\,x - 13.2793161162694)}$$

```
> r(1804):=evalf(subs(x=1804,r(x)));
```

$$r(1804) := 1.$$

```
> r(1804)[data]:=1;
```

$$r(1804)_{data} := 1$$

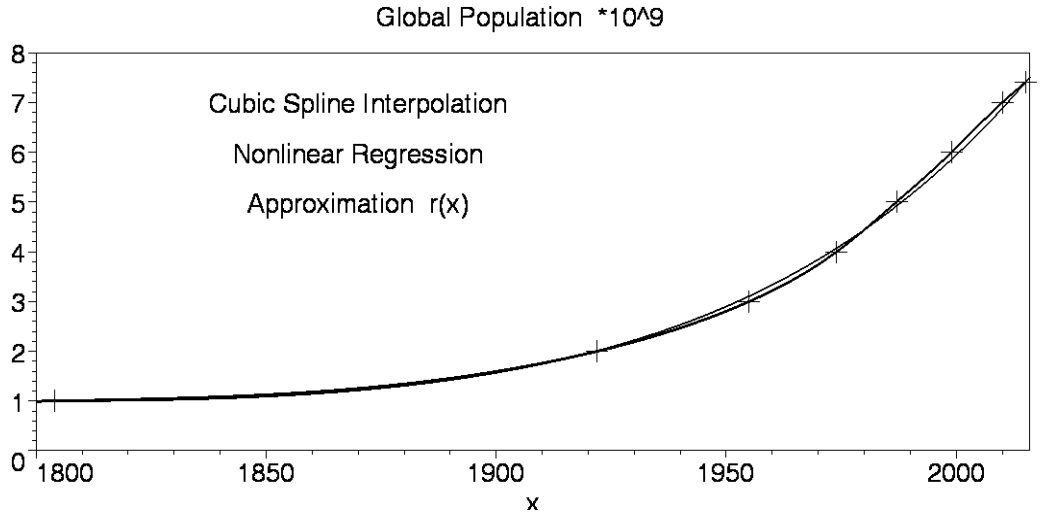
```
> r(2015):=evalf(subs(x=2015,r(x)));
```

$$r(2015) := 7.39999999975997$$

```
> r(2015)[data]:=7.4;
```

$$r(2015)_{data} := 7.4$$

```
>
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1800..2016,0..8,axes=boxed,
  th=3,co=black,style=point,symbol=cross,symbolsize=45):
> p[2]:=plot(Sp(x),x=1800..2015,th=3,co=black,
  title="Global Population  *10^9"):
> p[3]:=plot(r(x),x=1800..2016,th=2,co=black):
> p[4]:=plots[textplot]({[1870,7,`Cubic Spline Interpolation`],
  [1870,6,`Nonlinear Regression`],[1870,5,`Approximation
  r(x)`]}):
> plots[display](seq(p[k],k=1..4));
```



The N[2] *error norm* between the cubic spline interpolation and the approximation r(x) can be expressed as:

```
> N[2]:=
```

```
  sqrt((1/211)*Int((SPLINE - R(xi)),xi=1804..2015))=
  evalf(sqrt((1/211)*int((Sp(x)-r(x))^2,x=1804..2015)));
>
```

$$N_2 := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} SPLINE - \mathrm{R}(\xi)\, d\xi} = 0.06618117457$$

The n[2] *error norm* of the approximation r(x) with respect to the data is given by:

```
> with(linalg):
> for i from 1 to 8 do
  q[i]:=evalf(subs(x=DATA[i][1],r(x))-DATA[i][2]) od:
> Q:=vector([seq(q[i],i=1..8)]);
```

$$Q := [\,0.,\ 0.00941833345058884,\ 0.107450662525681,\ 0.0721448291481095,$$
$$-0.0825344101343100,\ -0.141138248878021,\ -0.120164049584496,$$
$$-0.240033770637638\ 10^{-9}\,]$$

```
> n[2]:=
  (1/sqrt(number_of_points))*Norm(Q,2)=
  evalf((1/sqrt(8))*norm(Q,2));
```

$$n_2 := \frac{\mathrm{Norm}(Q, 2)}{\sqrt{number\_of\_points}} = 0.0851546821888804$$

The calculated *error norms* { L[2], l[2] }, { M[2], m[2] }, and { N[2], n[2] } show that, respectively, the approximations y(x), z(x), and r(x) are suitable model functions to the given data.

Note: In this worksheet we have *evaluated* the approximations *using floating-point arithmetik* **(evalf)** with *high precision*, with Digits = default. For Digits < ten, e.g. equal to four, we arrive at nicer looking formulae. However, the results, e.g. the calculated *error norms* or the *boundary conditions* could be wrong! Let's show some examples:

```
> y(x)[DIGITS=4]:=evalf(y(x),4);
```

$$\mathrm{y}(x)_{DIGITS=4} := 0.7807 + 0.1575\ \mathbf{e}^{(0.01782\,x - 32.14)}$$

```
> y(1804)[DIGITS=4]:=evalf(subs(x=1804,y(x)[DIGITS=4]),4);
```

$$\mathrm{y}(1804)_{DIGITS=4} := 0.9398$$

```
> y(1804)[data]:=1;
```

$$\mathrm{y}(1804)_{data} := 1$$

```
> y(2015)[DIGITS=4]:=evalf(subs(x=2015,y(x)[DIGITS=4]),4);
```

$$\mathrm{y}(2015)_{DIGITS=4} := 7.613$$

```
> y(2015)[data]:=7.4;
```

$$\mathrm{y}(2015)_{data} := 7.4$$

```
> L[2][DIGITS=4]:=
```

```
sqrt((1/211)*Int((SPLINE-APPROX)^2,x=1804..2015))=
evalf(sqrt((1/211)*int((Sp(x)-y(x)[DIGITS=4])^2,x=1804..2015)),4
);
```

$$L_{2}{}_{DIGITS=4} := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} (SPLINE - APPROX)^2\, dx} = 0.07450$$

```
> L[2][DIGITS=default]:=0.07131679787;
```

$$L_{2}{}_{DIGITS=default} := 0.07131679787$$

```
> with(linalg):
> for i from 1 to 8 do
  v_[i]:=evalf(subs(x=DATA[i][1],y(x)[DIGITS=4])-DATA[i][2]) od:
> V_:=vector([seq(v_[i],i=1..8)]);
```

$$V\_ := [-0.0606492162,\ 0.079849967,\ 0.119809215,\ 0.062362583,\ -0.082137447,\ -0.095736570,\ 0.013777423,\ 0.194637721]$$

```
> l[2][DIGITS=4]:=
  (1/sqrt(number_of_points))*Norm(V_,2)=
  evalf((1/sqrt(8))*norm(V_,2),4);
```

$$l_{2}{}_{DIGITS=4} := \frac{\mathrm{Norm}(V\_, 2)}{\sqrt{number\_of\_points}} = 0.1014$$

```
> l[2][DIGITS=default]:=0.0955846901570484;
```

$$l_{2}{}_{DIGITS=default} := 0.0955846901570484$$

The next example is concerned with z(x):

```
> z(x)[DIGITS=4]:=evalf(z(x),4);
```

$$z(x)_{DIGITS=4} := 0.8356 + 0.1644\, \mathbf{e}^{(0.01748\, x - 31.53)}$$

```
> z(1804)[DIGITS=4]:=evalf(subs(x=1804,z(x)[DIGITS=4]));
```

$$z(1804)_{DIGITS=4} := 1.000645713$$

```
> z(1804)[data]:=1;
```

$$z(1804)_{data} := 1$$

```
> z(2015)[DIGITS=4]:=evalf(subs(x=2015,z(x)[DIGITS=4]));
```

$$z(2015)_{DIGITS=4} := 7.433472203$$

```
> z(2015)[data]:=7.4;
```

$$z(2015)_{data} := 7.4$$

```
> M[2][DIGITS=4]:=
  sqrt((1/211)*Int((SPLINE-APPROX)^2,x=1804..2015))=
  evalf(sqrt((1/211)*int((Sp(x)-z(x)[DIGITS=4])^2,x=1804..2015)),4
  );
```

$$M_{2}{}_{DIGITS=4} := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} (SPLINE - APPROX)^2\, dx} = 0.1131$$

```
> M[2][DIGITS=default]:=0.1149950158;
```

$$M_{2}{}_{DIGITS = default} := 0.1149950158$$

Another example has been concerned with the approximation r(x):

```
> r(x)[DIGITS=4]:=evalf(r(x),4);
```

$$r(x)_{DIGITS = 4} := 1. + 1.354\,(0.004739\,x - 8.550)^{2}\,\mathbf{e}^{(0.007361\,x - 13.28)}$$

```
> r(1804)[DIGITS=4]:=evalf(subs(x=1804,r(x)[DIGITS=4]),4);
```

$$r(1804)_{DIGITS = 4} := 1.$$

```
> r(1804)[data]:=1;
```

$$r(1804)_{data} := 1$$

```
> r(2015)[DIGITS=4]:=evalf(subs(x=2015,r(x)[DIGITS=4]),4);
```

$$r(2015)_{DIGITS = 4} := 7.365$$

```
> r(2015)[data]:=7.4;
```

$$r(2015)_{data} := 7.4$$

```
> N[2][DIGITS=4]:=
  sqrt((1/211)*Int((SPLINE-APPROX)^2,x=1804..2015))=
  evalf(sqrt((1/211)*int((Sp(x)-r(x)[DIGITS=4])^2,x=1804..2015)),4
  );
```

$$N_{2}{}_{DIGITS = 4} := \frac{1}{211}\sqrt{211}\sqrt{\int_{1804}^{2015} (SPLINE - APPROX)^{2}\,dx} = 0.1196$$

```
> N[2][DIGITS=10]:=0.06809235434;
```

$$N_{2}{}_{DIGITS = 10} := 0.06809235434$$

```
> with(linalg):
> for i from 1 to 8 do
  q_[i]:=evalf(subs(x=DATA[i][1],r(x)[DIGITS=4])-DATA[i][2],4) od:
> Q_:=vector([seq(q_[i],i=1..8)]);
```

$$Q\_ := [0.1354\,10^{-5}, 0.006, 0.100, 0.062, -0.081, -0.182, -0.116, -0.035]$$

```
> n[2][DIGITS=4]:=
  (1/sqrt(number_of_points))*Norm(Q_,2)=
  evalf((1/sqrt(8))*norm(Q_,2),4);
```

$$n_{2}{}_{DIGITS = 4} := \frac{\mathrm{Norm}(Q\_, 2)}{\sqrt{number\_of\_points}} = 0.09233$$

```
> n[2][DIGITS=default]:=0.0913215202892018;
```

$$n_{2}{}_{DIGITS = default} := 0.0913215202892018$$

```
>
```

The above comparisons show that there are some differences between Digits = 4 and Digits = 10 = default!

Appendix

In the following the development of the global population from t = 1500 to 2015 has been calculated.

```
> restart:
> with(Statistics):
> DATA:=
  [1500,0.5],[1750,0.60],[1800,0.70],[1850,0.90],
  [1900,1.50],[1950,2.75],[1960,3.20],[1970,3.68],
  [1980,4.44],[1990,5.31],[2000,6.13],[2005,6.52],
  [2010,6.93],[2015,7.4];
```

DATA := [1500, 0.5], [1750, 0.60], [1800, 0.70], [1850, 0.90], [1900, 1.50], [1950, 2.75], [1960, 3.20], [1970, 3.68], [1980, 4.44], [1990, 5.31], [2000, 6.13], [2005, 6.52], [2010, 6.93], [2015, 7.4]

```
> X:=array([1500,1750,1800,1850,1900,1950,
  1960,1970,1980,1990,1000,2005,2010,2015]);
```

X := [1500, 1750, 1800, 1850, 1900, 1950, 1960, 1970, 1980, 1990, 1000, 2005, 2010, 2015]

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

symbol

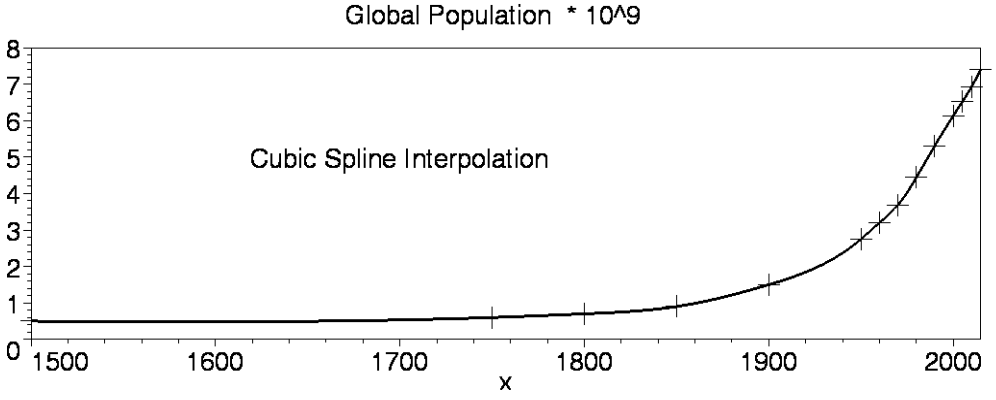
```
> Y:=array([0.5,0.60,0.70,0.90,1.50,2.75,3.20,
  3.68,4.44,5.31,6.13,6.52,6.93,7.4]);
```

Y := [0.5, 0.60, 0.70, 0.90, 1.50, 2.75, 3.20, 3.68, 4.44, 5.31, 6.13, 6.52, 6.93, 7.4]

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

symbol

```
> with(CurveFitting):
> Sp(x):=Spline([DATA],x,degree=3):
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1500..2015,0..8,th=3,co=black,
  style=point,symbol=cross,symbolsize=45,axes=boxed,
  title="Global Population  * 10^9"):
> p[2]:=plot(Sp(x),x=1500..2015,th=3,co=black):
> p[3]:=plots[textplot]([1700,5,`Cubic Spline Interpolation`]):
> plots[display](seq(p[k],k=1..3));
```



Exponential Growth with factor [(t-1000)/1015)^2]^2 as an example:

```
> h:=(t,a) ->
  0.5+(6.9/exp(a))*((t-1500)/515)^2*exp(a*(t-1500)/515);
```

$$h := (t, a) \to 0.5 + \frac{6.9 \left(\frac{1}{515} t - \frac{300}{103} \right)^2 \mathbf{e}^{(1/515\, a\, (t - 1500))}}{\mathbf{e}^{a}}$$

```
> R:=unapply(evalf(NonlinearFit(h(t,a),X,Y,t)),t);
```

$$R := t \to 0.5 + 0.0117211428310179\, (0.001941747573\, t - 2.912621359)^2\, \mathbf{e}^{(0.012384237669925\, t - 18.5763565048875)}$$

```
> R(x):=subs(t=x,R(t));
```

$$R(x) := 0.5 + 0.0117211428310179\, (0.001941747573\, x - 2.912621359)^2\, \mathbf{e}^{(0.0123842376699250\, x - 18.5763565048875)}$$

```
> R(1500):=evalf(subs(x=1500,R(x)));
```

$$R(1500) := 0.500000000000000$$

```
> R(1500)[data]:=0.5;
```

$$R(1500)_{data} := 0.5$$

```
> R(2015):=evalf(subs(x=2015,R(x)));
```

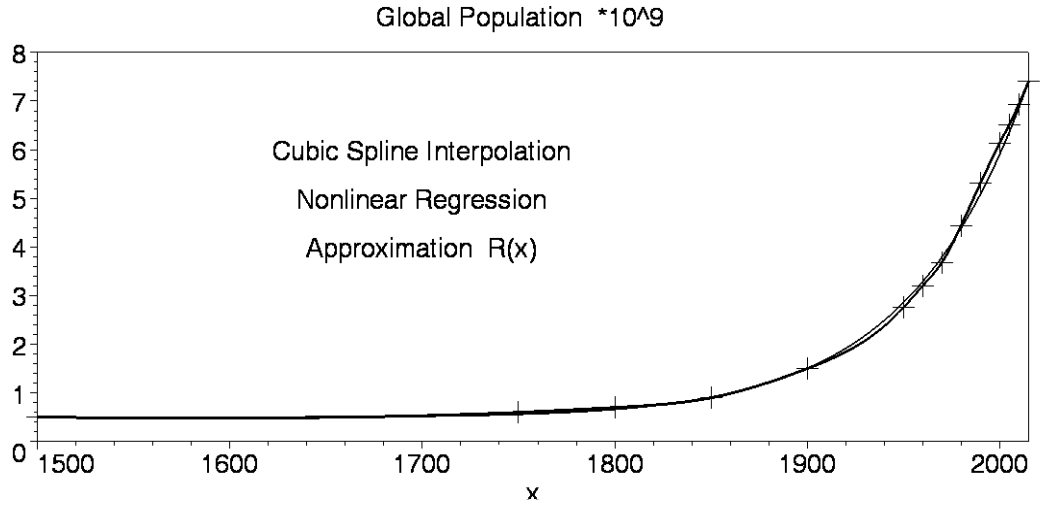
$$R(2015) := 7.40000001375334$$

```
> R(2015)[data]:=7.4;
```

$$R(2015)_{data} := 7.4$$

The boundary conditions are fulfilled exactly.

```
> alias(th=thickness,co=color):
> p[1]:=plot([DATA],x=1500..2015,0..8,axes=boxed,th=3,
  co=black,style=point,symbol=cross,symbolsize=45):
> p[2]:=plot(Sp(x),x=1500..2015,th=3,
  co=black,title="Global Population  *10^9"):
> p[3]:=plot(R(x),x=1500..2015,th=2,co=black):
> p[4]:=plots[textplot]({[1700,6,`Cubic Spline Interpolation`],
  [1700,5,`Nonlinear Regression`],[1700,4,`Approximation
  R(x)`]}):
> plots[display](seq(p[k],k=1..4));
```



The L[2] *error norm between* the cubic spline interpolation and the approximation R(x) can be expressed as:

```
> L[2]:=
  sqrt((1/515)*Int((SPLINE -Approximation)^2,x=1500..2015))=
  evalf(sqrt((1/515)*int((Sp(x)-R(x))^2,x=1500..2015)));
```

$$L_2 := \frac{1}{515}\sqrt{515}\sqrt{\int_{1500}^{2015} (SPLINE - Approximation)^2\, dx} = 0.06103774163$$

The l[2] *error norm* of the approximation R(x) with respect to the data is given by:

```
> with(linalg):
> for i from 1 to 14 do
  q[i]:=evalf(subs(x=DATA[i][1],R(x))-DATA[i][2]) od:
> Q:=vector([seq(q[i],i=1..14)]);
```

$$Q := [0.117211428310179\ 10^{-19}, -0.0389287176067251, -0.0366497382960533, 0.0129844679111111, 0.00193123738315393, 0.105392376097047, 0.0857232060100381, 0.111562269769442, -0.0542675383081299, -0.226822882193821, -0.228696877736548, -0.158168522563107, -0.0696215832530589, 0.137533389121813\ 10^{-7}]$$

```
> l[2]:=
  (1/sqrt(number_of_points))*Norm(Q,2)=
  evalf((1/sqrt(14))*norm(Q,2));
```

$$l_2 := \frac{\mathrm{Norm}(Q, 2)}{\sqrt{number\_of\_points}} = 0.110354438506378$$

Note: The above approximation R(x) and the error norms L[2], l[2] have been calculated with "high precision" , that is,
with Digits = default = 10. We should compare these results with Digits = 4, for instance:

```
> R(x)[DIGITS=4]:=evalf(R(x),4);
```

$$R(x)_{DIGITS=4} := 0.5 + 0.01172\,(0.001942\,x - 2.913)^2\,\mathbf{e}^{(0.01238\,x - 18.58)}$$

```
> R(1500)[DIGITS=4]:=evalf(subs(x=1500,R(x)[DIGITS=4]),4);
```

$$R(1500)_{DIGITS=4} := 0.5$$

```
> R(1500)[data]:=0.5;
```

$$R(1500)_{data} := 0.5$$

```
> R(2015)[DIGITS=4]:=evalf(subs(x=2015,R(x)[DIGITS=4]),4);
```

$$R(2015)_{DIGITS=4} := 7.346$$

```
> R(2015)[data]:=7.4;
```

$$R(2015)_{data} := 7.4$$

```
> L[2][DIGITS=4]:=
  sqrt((1/515)*Int((SPLINE-APPROX)^2,x=1500..2015))=
  evalf(sqrt((1/515)*int((Sp(x)-R(x)[DIGITS=4])^2,x=1500..2015)),4
  );
```

$$L_{2_{DIGITS=4}} := \frac{1}{515}\sqrt{515}\sqrt{\int_{1500}^{2015} (SPLINE - APPROX)^2\,dx} = 0.06967$$

```
> L[2][DIGITS=10]:=0.06103774163;
```

$$L_{2_{DIGITS=10}} := 0.06103774163$$

```
> with(linalg):
> for i from 1 to 14 do
  q_[i]:=evalf(subs(x=DATA[i][1],R(x)[DIGITS=4])-DATA[i][2]) od:
> Q_:=vector([seq(q_[i],i=1..14)]);
```

$$Q\_ := [0.,\ -0.03959059519,\ -0.0384543258,\ 0.0083355549,\ -0.0095572551,\ 0.077891337,\ 0.053081049,\ 0.072855013,\ -0.100124690,\ -0.281102785,\ -0.292892263,\ -0.227960091,\ -0.145482063,\ -0.082441010]$$

```
> l[2][DIGITS=4]:=
  (1/sqrt(number_of_points))*Norm(Q_,2)=
  evalf((1/sqrt(14))*norm(Q_,2),4);
```

$$l_{2_{DIGITS=4}} := \frac{\mathrm{Norm}(Q\_, 2)}{\sqrt{number\_of\_points}} = 0.1394$$

```
> l[2][DIGITS=10]:=0.148135737630170;
```

$$l_{2_{DIGITS=10}} := 0.148135737630170$$

We see, the comparisons show that there are some differences between digits = 4 and digits = default.

Prognosis for future developments of the *global population *10^9* assuming the above approximation R(x) in the period from 2015 to 2050, too:

```
> for i in [2015,2025,2035,2050] do
  R(i):=evalf(subs(t=i,R(t)),4) od;
```

$$R(2015) := 7.346$$

$$R(2025) := 8.527$$

$$R(2035) := 9.900$$

$$R(2050) := 12.50$$

We see, within a period of only 35 years the global population increases by 5*10^9. In contrast: Within the period from 1800 to 1980 , i. e. within 180 years, the global population has increased by 5*10^9, too.

>