Kurs Komputerowy S
System Symboliczny
Mathematica

Obliczenia numeryczne

Dokladnosc i precyzja

Precision[wartosc]
SetPrecision[wartosc, precyzja]
Accuracy[wartosc]
SetAccuracy[wartosc, dokladnosc]
MachinePrecision

In[1]:=
```
a = Sqrt[2]
```

Out[1]=
$$\sqrt{2}$$

In[2]:=
```
Precision[a]
```

Out[2]=
$$\infty$$

In[3]:=
```
Precision[1.3]
```

Out[3]=
```
MachinePrecision
```

In[4]:=
```
b = 1.3`50
```
Out[4]=
```
1.3000000000000000000000000000000000000000000000000000
```

In[5]:=
```
Precision[b]
```
Out[5]=
```
50.
```

In[6]:=
```
b = Sqrt[2.]
```
Out[6]=
```
1.41421
```

In[7]:=
```
Precision[b]
```
Out[7]=
```
MachinePrecision
```

In[8]:=
```
$MachinePrecision
```
Out[8]=
```
15.9546
```

In[9]:=
```
c = SetPrecision[b, 20]
```
Out[9]=
```
1.4142135623730951455
```

In[10]:=
```
Precision[c]
```
Out[10]=
```
20.
```

In[11]:=
```
Accuracy[c]
```
Out[11]=
```
19.8495
```

In[12]:=
```
Accuracy[1.3]
```
Out[12]=
```
15.8406
```

In[13]:=
```
d = 1.3‘‘50
```

Out[13]=
```
1.300000000000000000000000000000000000000000000000000
```

In[14]:=
```
Accuracy[d]
```

Out[14]=
```
50.
```

■ Wynik dokladny a przyblizony

wyrazenie // N
N[wyrazenie]
N[wyrazenie, precyzja]
Chop[wyrazenie]

In[15]:=
```
a = Sqrt[2]
```

Out[15]=
$$\sqrt{2}$$

In[16]:=
```
b = Sqrt[2.]
```

Out[16]=
```
1.41421
```

In[17]:=
```
a // N
```

Out[17]=
```
1.41421
```

In[18]:=
```
Precision[%]
```

Out[18]=
```
MachinePrecision
```

In[19]:=
```
N[a, 50]
```

Out[19]=
```
1.4142135623730950488016887242096980785696718753769
```

In[20]:=
```
Precision[%]
```
Out[20]=
```
50.
```

In[21]:=
```
Chop[1.3]
```
Out[21]=
```
1.3
```

In[22]:=
```
Chop[0.90 * 10^-10]
```
Out[22]=
```
0
```

In[23]:=
```
Chop[1.0 * 10^-20]
```
Out[23]=
```
0
```

In[24]:=
```
1.0 * 10^-20
```
Out[24]=
$$1. \times 10^{-20}$$

In[25]:=
```
Fourier[{1, 0, 3, 0, 7, 0, 6, 4, 1}]
```
Out[25]=
$$\{7.33333 + 0.\,i,\ -2.19876 - 1.47654\,i,\ -1.01396 - 0.210063\,i,\ -0.166667 + 2.02073\,i,\ 1.21272 - 3.92968\,i,\ 1.21272 + 3.92968\,i,\ -0.166667 - 2.02073\,i,\ -1.01396 + 0.210063\,i,\ -2.19876 + 1.47654\,i\}$$

In[26]:=
```
InverseFourier[%]
```
Out[26]=
$$\{1.,\ 5.92119 \times 10^{-16},\ 3.,\ 0.,\ 7.,\ 7.40149 \times 10^{-16},\ 6.,\ 4.,\ 1.\}$$

In[27]:=
```
Chop[%]
```
Out[27]=
```
{1., 0, 3., 0, 7., 0, 6., 4., 1.}
```

■ Suma i iloczyn elementow ciagu

NSum[wyrazenie, {zmienna, w_pocz, w_konc}]
NProduct[wyrazenie, {zmienna, w_pocz, w_konc}]

Options[polecenie]

In[28]:=
```
NSum[1 / x^2, {x, 1, 100}]
```

Out[28]=
```
1.63498
```

In[29]:=
```
NSum[1 / x^2, {x, 1, Infinity}]
```

Out[29]=
```
1.64493
```

In[30]:=
```
Sum[1 / x^2, {x, 1, Infinity}]
```

Out[30]=
$$\frac{\pi^2}{6}$$

In[31]:=
```
N[%]
```

Out[31]=
```
1.64493
```

In[32]:=
```
Options[NSum]
```

Out[32]=
```
{AccuracyGoal → ∞, Compiled → Automatic, EvaluationMonitor → None,
 Method → Automatic, NSumTerms → 15, PrecisionGoal → Automatic,
 VerifyConvergence → True, WorkingPrecision → MachinePrecision}
```

In[33]:=
```
NSum[1 / x^2, {x, 1, Infinity}, WorkingPrecision -> 50]
```

Out[33]=
```
1.6449340668482264364724151625561175644300316423611
```

In[34]:=
```
NSum[1 / x^2, {x, 1, Infinity}, PrecisionGoal -> 50, WorkingPrecision -> 20]
```

NIntegrate::ncvb : NIntegrate failed to converge to prescribed accuracy after
9 recursive bisections in x near {x} = {30.517807598914247980}. NIntegrate obtained
0.0625'20. and 9.402357391154246138'20.*^-23 for the integral and error estimates. >>

Out[34]=
```
1.6449340668482264365
```

In[35]:=
```
Precision[%]
```

Out[35]=
```
20.
```

In[36]:=
```
NSum[1 / x^2, {x, 1, 10},
 EvaluationMonitor :> Print["x=", x, "    ", N[1 / x^2, 10]]]
```

```
x=1.    1.
x=2.    0.25
x=3.    0.111111
x=4.    0.0625
x=5.    0.04
x=6.    0.0277778
x=7.    0.0204082
x=8.    0.015625
x=9.    0.0123457
x=10.    0.01
```

Out[36]=
```
1.54977
```

In[37]:=
```
NSum[1 / x^2, {x, 1, 10}, WorkingPrecision -> 50,
 EvaluationMonitor :> Print["x=", x, "    ", N[1 / x^2, 10]]]
```

```
x=1.00000000000000000000000000000000000000000000000000    1.000000000
x=2.00000000000000000000000000000000000000000000000000    0.2500000000
x=3.00000000000000000000000000000000000000000000000000    0.1111111111
x=4.00000000000000000000000000000000000000000000000000    0.06250000000
x=5.00000000000000000000000000000000000000000000000000    0.04000000000
x=6.00000000000000000000000000000000000000000000000000    0.02777777778
x=7.00000000000000000000000000000000000000000000000000    0.02040816327
x=8.00000000000000000000000000000000000000000000000000    0.01562500000
x=9.00000000000000000000000000000000000000000000000000    0.01234567901
x=10.0000000000000000000000000000000000000000000000000    0.01000000000
```

Out[37]=
```
1.549767731166540690350214159737969261778785588309 4
```

In[38]:=
```
NProduct[x / (x + 1), {x, 1, 100}]
```

Out[38]=
```
0.00990099
```

In[39]:=
```
Product[x / (x + 1), {x, 1, 100}]
```

Out[39]=
$$\frac{1}{101}$$

In[40]:=
```
N[%]
```

Out[40]=
```
0.00990099
```

In[41]:=
```
Options[NProduct]
```

Out[41]=
```
{AccuracyGoal → ∞, Compiled → Automatic, EvaluationMonitor → None,
 Method → Automatic, NProductFactors → 15, PrecisionGoal → Automatic,
 VerifyConvergence → True, WorkingPrecision → MachinePrecision}
```

■ Rozwiazywanie rownan

NSolve[rown, zmienna] (* NSolve[...]≡N[Solve[...]] *)

In[42]:=
```
eq = x^2 + 4 x - 3 == 0
NSolve[eq, x]
```

Out[42]=
$$-3 + 4\,x + x^2 == 0$$

Out[43]=
$$\{\{x \to -4.64575\}, \{x \to 0.645751\}\}$$

In[44]:=
```
Solve[eq, x]
```

Out[44]=
$$\left\{\left\{x \to -2 - \sqrt{7}\right\}, \left\{x \to -2 + \sqrt{7}\right\}\right\}$$

In[45]:=
```
N[%]
```

Out[45]=
$$\{\{x \to -4.64575\}, \{x \to 0.645751\}\}$$

In[46]:=
```
Clear[a, b, c]
```

In[47]:=
```
NSolve[a x^2 + b x + c == 0, x]
```

Out[47]=
$$\left\{\left\{x \to \frac{0.5\left(-1.\,b - 1.\sqrt{b^2 - 4.\,a\,c}\right)}{a}\right\}, \left\{x \to \frac{0.5\left(-1.\,b + \sqrt{b^2 - 4.\,a\,c}\right)}{a}\right\}\right\}$$

FindRoot[wielomian, zmienna]

In[48]:=
```
f = x^4 - 4 x^3 - 2 x^2 + 5 x + 1
FindRoot[f, {x, 1}]
```

Out[48]=
```
1 + 5 x - 2 x^2 - 4 x^3 + x^4
```

Out[49]=
```
{x → 1.12503}
```

In[50]:=
```
FindRoot[f, {x, 0}]
```

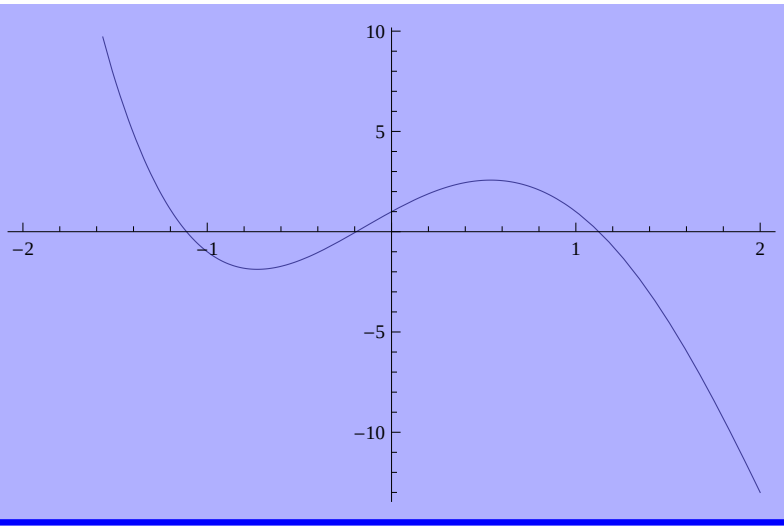
Out[50]=
```
{x → -0.191234}
```

In[51]:=
```
FindRoot[f, {x, -1}]
```

Out[51]=
```
{x → -1.11236}
```

In[52]:=
```
Plot[f, {x, -2, 2}]
```

Out[52]=



■ Interpolacja, ekstrapolacja i aproksymacja

Interpolation[dane]
  dane:
    {f1, f2, f3, ...}
    {{x1, f1}, {x2, f2}, {x3, f3}, ...}
    {{{x1, y2, ...}, f1}, {{x2, y2, ...}, f2}, ...}
Interpolation[dane,wartosc]

In[53]:=
```
d = {1, 5, 2, 6, 3, 7, 4, 8}
fi = Interpolation[d]
```

Out[53]=
```
{1, 5, 2, 6, 3, 7, 4, 8}
```

Out[54]=
```
InterpolatingFunction[{{1, 8}}, <>]
```

In[55]:=
```
fi[2]
fi[2.5]
```

Out[55]=
```
5
```

Out[56]=
```
3.5
```

In[57]:=
```
fi[10]
```

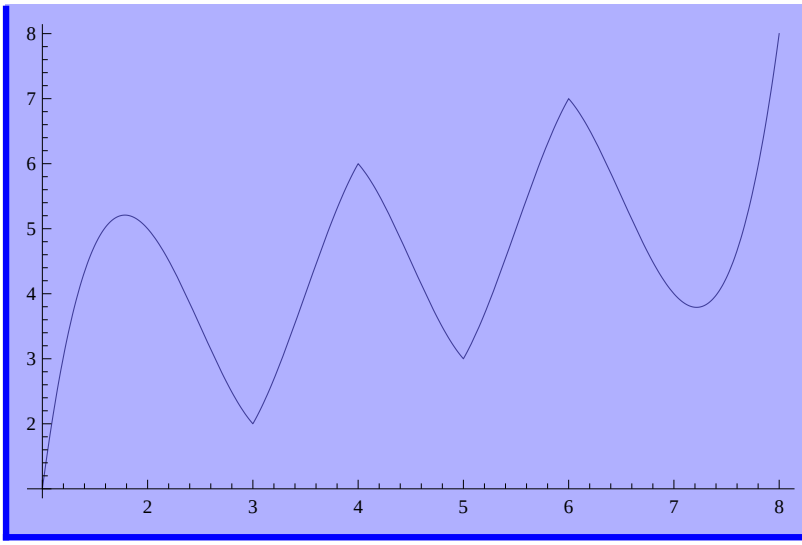
InterpolatingFunction::dmval :

Input value {10} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

Out[57]=
```
93
```

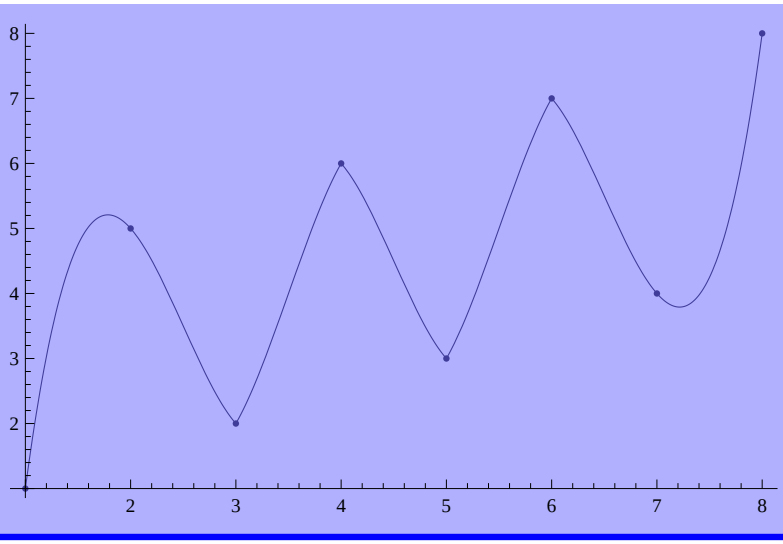
In[58]:=
```
Plot[fi[x], {x, 1, 8}]
```

Out[58]=



In[59]:=
```
Show[%, ListPlot[d]]
```

Out[59]=



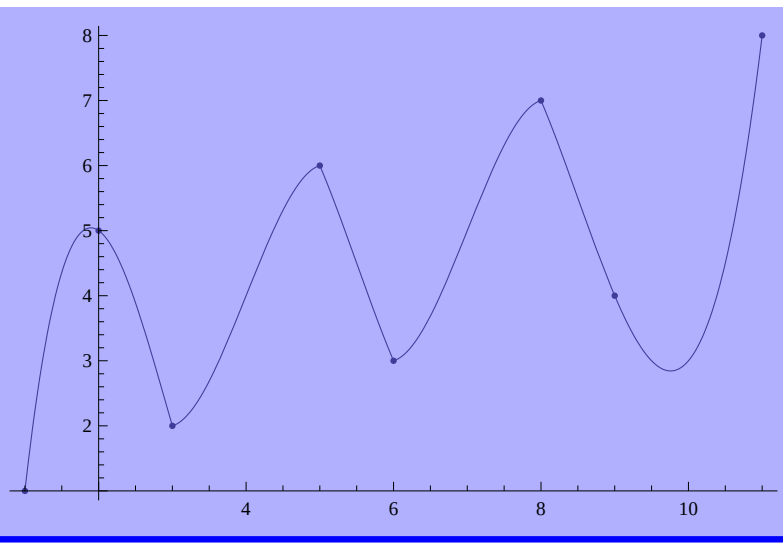
In[60]:=
```
d = {{1, 1}, {2, 5}, {3, 2}, {5, 6}, {6, 3}, {8, 7}, {9, 4}, {11, 8}}
fi = Interpolation[d]
```

Out[60]=
```
{{1, 1}, {2, 5}, {3, 2}, {5, 6}, {6, 3}, {8, 7}, {9, 4}, {11, 8}}
```

Out[61]=
```
InterpolatingFunction[{{1, 11}}, <>]
```

In[62]:=
```
Show[Plot[fi[x], {x, 1, 11}], ListPlot[d]]
```

Out[62]=



In[63]:=
```
Options[Interpolation]
```

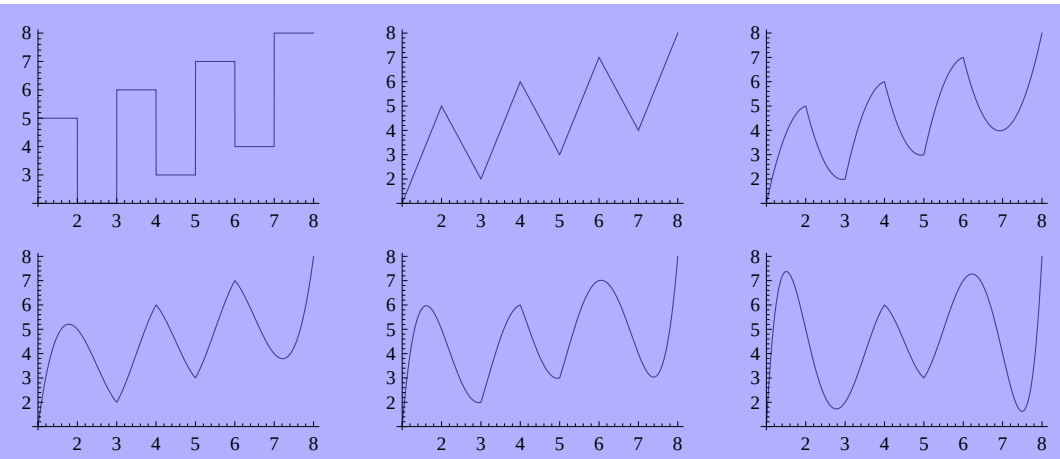
Out[63]=
```
{InterpolationOrder → 3, Method → Automatic, PeriodicInterpolation → False}
```

In[64]:=
```
d = {1, 5, 2, 6, 3, 7, 4, 8}
f0 = Interpolation[d, InterpolationOrder -> 0];
f1 = Interpolation[d, InterpolationOrder -> 1];
f2 = Interpolation[d, InterpolationOrder -> 2];
f3 = Interpolation[d, InterpolationOrder -> 3];
f4 = Interpolation[d, InterpolationOrder -> 4];
f5 = Interpolation[d, InterpolationOrder -> 5];
```

Out[64]=
```
{1, 5, 2, 6, 3, 7, 4, 8}
```

In[71]:=
```
GraphicsGrid[
 {{Plot[f0[x], {x, 1, 8}], Plot[f1[x], {x, 1, 8}], Plot[f2[x], {x, 1, 8}]},
  {Plot[f3[x], {x, 1, 8}], Plot[f4[x], {x, 1, 8}], Plot[f5[x], {x, 1, 8}]}}]
```

Out[71]=



Fit[dane, funkcja, zmienna]

In[72]:=
```
d = {1, 5, 2, 6, 3, 7, 4, 8};
f1 = Fit[d, {1, x}, x]
f2 = Fit[d, {1, x, x^2}, x]
f4 = Fit[d, {1, x, x^2, x^3, x^4}, x]
f7 = Fit[d, {1, x, x^2, x^3, x^4, x^5, x^6, x^7}, x]
```

Out[73]=
$$1.5 + 0.666667\,x$$

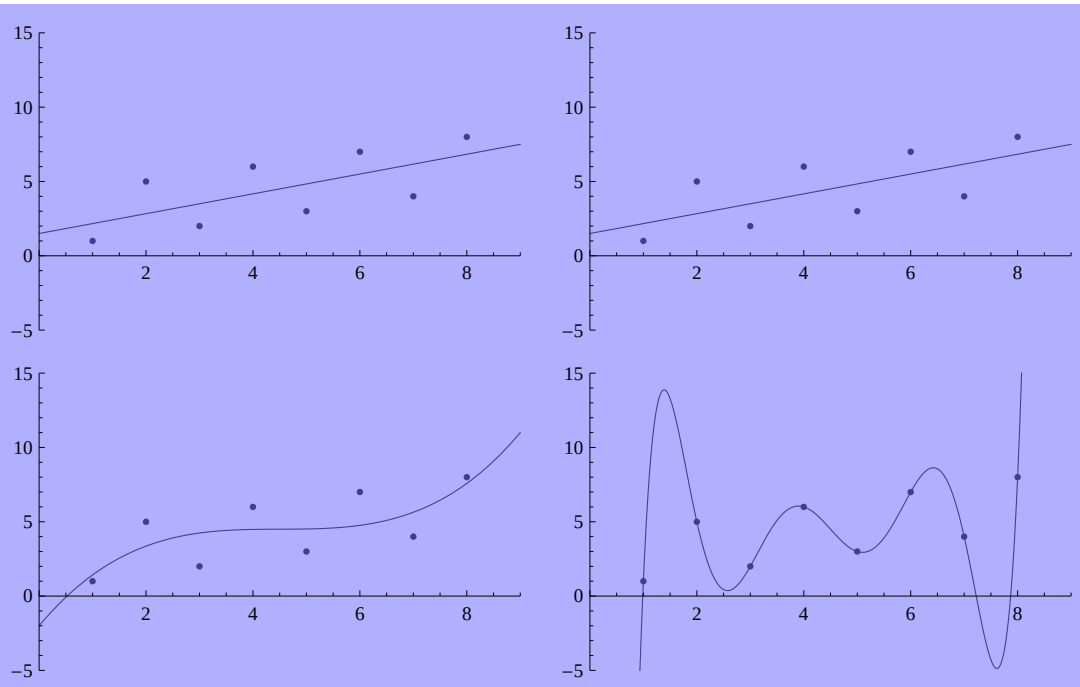
Out[74]=
$$1.5 + 0.666667\,x - 4.74156 \times 10^{-17}\,x^2$$

Out[75]=
$$-2. + 4.30808\,x - 0.954545\,x^2 + 0.0707071\,x^3 + 2.61 \times 10^{-16}\,x^4$$

Out[76]=
$$-444. + 1081.57\,x - 999.6\,x^2 + 468.844\,x^3 - 122.5\,x^4 + 18.0444\,x^5 - 1.4\,x^6 + 0.0444444\,x^7$$

In[77]:=
```
GraphicsGrid[
 {{Show[Plot[f1, {x, 0, 9}, PlotRange -> {{0, 9}, {-5, 15}}], ListPlot[d]],
   Show[Plot[f2, {x, 0, 9}, PlotRange -> {{0, 9}, {-5, 15}}], ListPlot[d]]},
  {Show[Plot[f4, {x, 0, 9}, PlotRange -> {{0, 9}, {-5, 15}}], ListPlot[d]],
   Show[Plot[f7, {x, 0, 9}, PlotRange -> {{0, 9}, {-5, 15}}], ListPlot[d]]}}]
```

Out[77]=



In[78]:=
```
Fit[{1}, {1, x, x^2}, x]
```

Out[78]=
$$0.333333 + 0.333333\,x + 0.333333\,x^2$$

FindFit[dane, funkcja, parametry, zmienna]
FindFit[dane,{funkcja,warunki},parametry,zmienna]

In[80]:=
```
Clear[f]
```

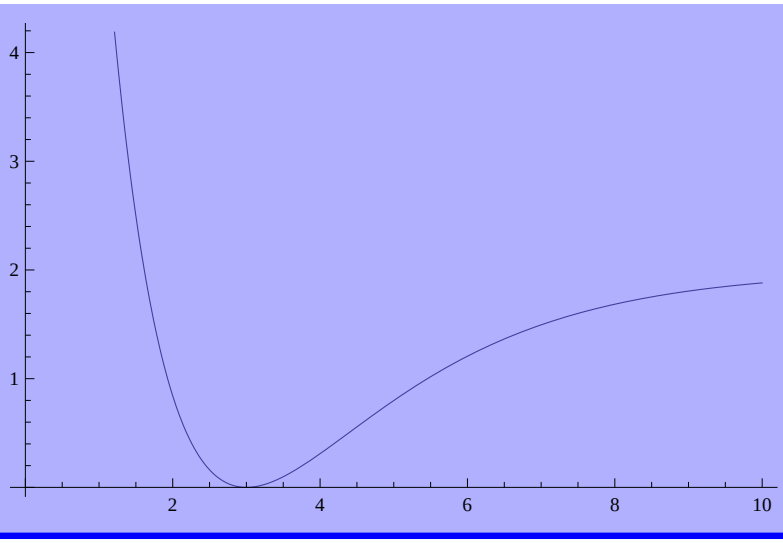
In[81]:=
```
f[De_, α_, xe_] := De (1 - Exp[-α (x - xe)])^2
```

In[82]:=
```
funkcja = f[2, 0.5, 3]
```

Out[82]=
$$2\left(1 - e^{-0.5\,(-3+x)}\right)^2$$

In[83]:=
```
Plot[funkcja, {x, 0, 10}]
```

Out[83]=



In[84]:=
```
dane = Table[{x, funkcja}, {x, 1, 10}]
```

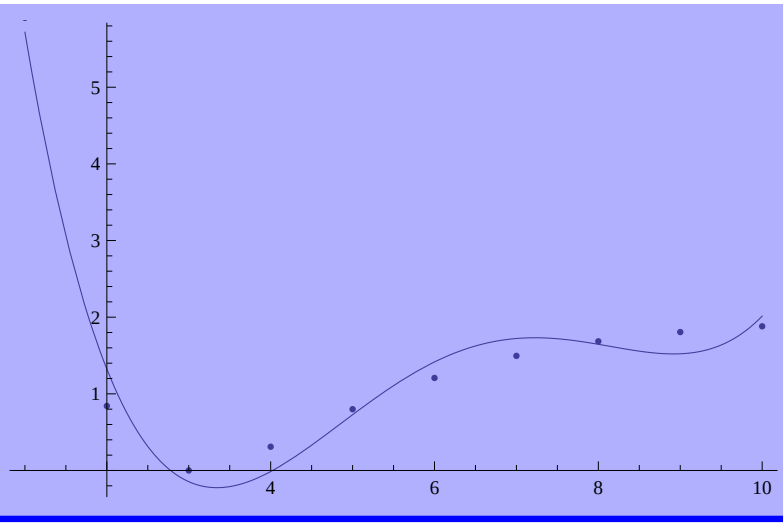
Out[84]=
```
{{1, 5.90498}, {2, 0.841679}, {3, 0.}, {4, 0.309636}, {5, 0.799153},
 {6, 1.20705}, {7, 1.49529}, {8, 1.68514}, {9, 1.80581}, {10, 1.88103}}
```

In[85]:=
```
ff = Fit[dane, {1, x, x^2, x^3, x^4}, x]
```

Out[85]=
$$14.6941 - 11.9016\,x + 3.2704\,x^2 - 0.35879\,x^3 + 0.0138087\,x^4$$

In[86]:=
```
Show[Plot[ff, {x, 1, 10}], ListPlot[dane]]
```

Out[86]=

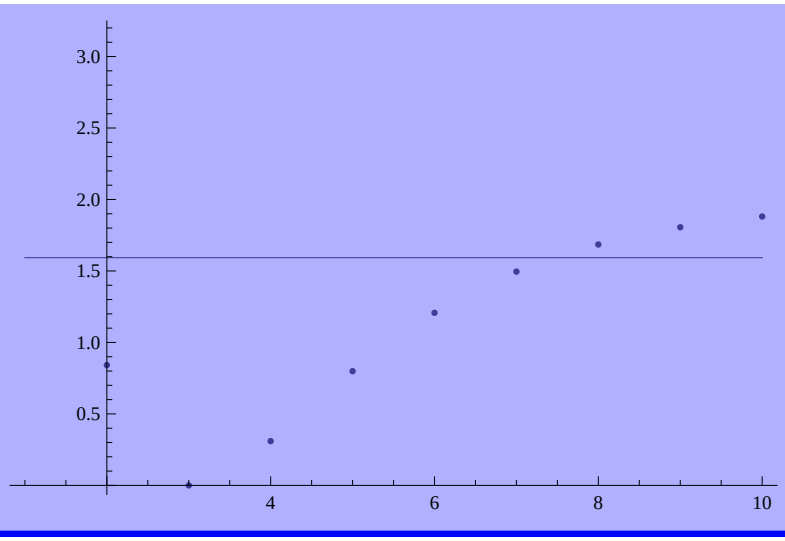


In[87]:=
```
ffit = FindFit[dane, a (1 - Exp[-b (x - c)])^2, {a, b, c}, x]
```

Out[87]=
```
{a → 1.59298, b → 2.11558, c → -41.8075}
```

In[88]:=
```
Show[Plot[f[a, b, c] /. ffit, {x, 1, 10}], ListPlot[dane]]
```

Out[88]=

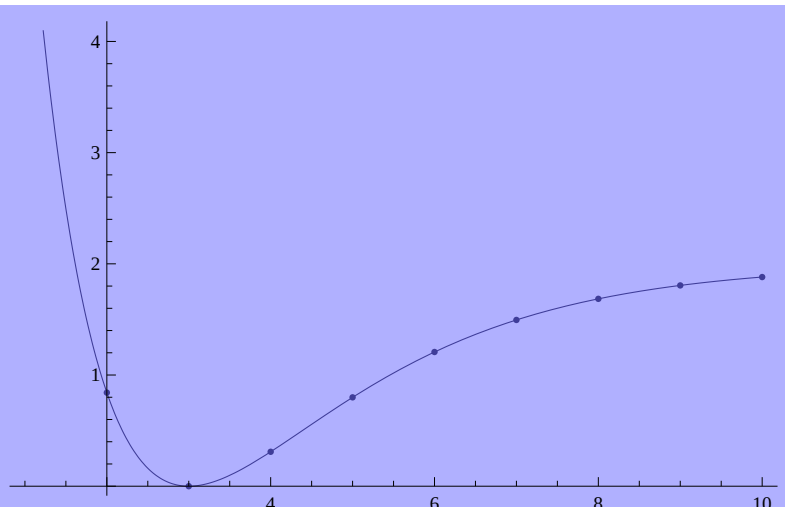


In[89]:=
```
ffit = FindFit[dane, {a (1 - Exp[-b (x - c)])^2, b > 0.3}, {a, b, c}, x]
```

Out[89]=
```
{a → 2.00023, b → 0.499938, c → 3.00016}
```

In[90]:=
```
Show[Plot[f[a, b, c] /. ffit, {x, 1, 10}], ListPlot[dane]]
```

Out[90]=



■ Minimum i maksimum funkcji

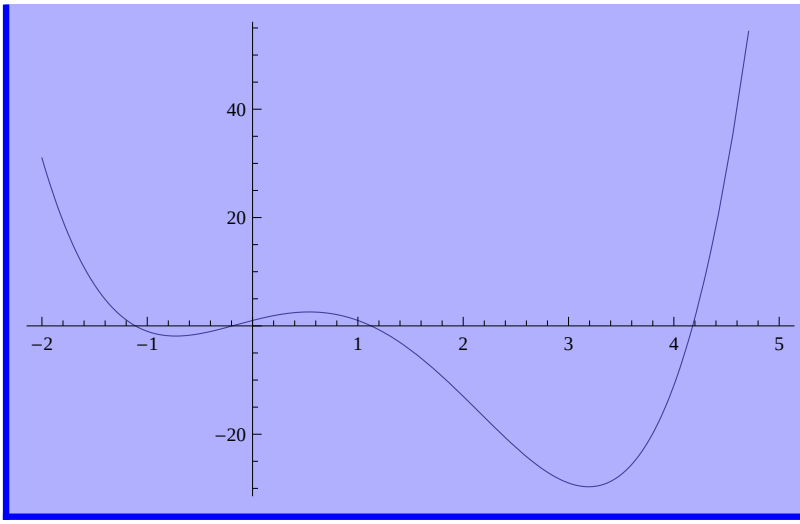
FindMinimum[funkcja, {zmienna, wartosc}]

NMinimize[funkcja,zmienna]
FindMaximum[funkcja,{zmienna,wartosc}]
NMaximize[funkcja,zmienna]
Opcje: StepMonitor i EvaluationMonitor

In[91]:=
```
f = x^4 - 4 x^3 - 2 x^2 + 5 x + 1
Plot[f, {x, -2, 5}]
```

Out[91]=
```
1 + 5 x - 2 x^2 - 4 x^3 + x^4
```

Out[92]=



In[93]:=
```
f = x^4 - 4 x^3 - 2 x^2 + 5 x + 1
FindMinimum[f, {x, 0}]
```

Out[93]=
```
1 + 5 x - 2 x^2 - 4 x^3 + x^4
```

FindMinimum::lstol :
The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances. >>

Out[94]=
```
{-1.87577, {x → -0.728447}}
```

In[95]:=
```
FindMinimum[f, {x, 3}]
```

Out[95]=
```
{-29.6962, {x → 3.19063}}
```

In[96]:=
```
FindMinimum[f, {x, 2}]
```

Out[96]=
```
{-29.6962, {x → 3.19063}}
```

In[97]:=
```
NMinimize[f, x]
```

Out[97]=
```
{-29.6962, {x → 3.19063}}
```

In[98]:=
```
Options[NMinimize]
```

Out[98]=
```
{AccuracyGoal → Automatic, EvaluationMonitor → None,
 MaxIterations → 100, Method → Automatic, PrecisionGoal → Automatic,
 StepMonitor → None, WorkingPrecision → MachinePrecision}
```

In[99]:=
```
NMinimize[f, x, Method → "DifferentialEvolution"]
```

Out[99]=
```
{-29.6962, {x → 3.19063}}
```

In[100]:=
```
FindMaximum[f, {x, 0}]
```

Out[100]=
```
{2.57201, {x → 0.537818}}
```

In[101]:=
```
FindMaximum[f, {x, -1}]
```

FindMaximum::cvmit : Failed to converge to the requested accuracy or precision within 100 iterations. ≫

Out[101]=
$$\{5.770242244403928 \times 10^{417}, \{x \to -2.75612 \times 10^{104}\}\}$$

In[102]:=
```
NMaximize[f, x]
```

NMaximize::cvdiv : Failed to converge to a solution. The function may be unbounded. ≫

Out[102]=
$$\{4.756968863562773 \times 10^{424}, \{x \to -1.47684 \times 10^{106}\}\}$$

In[103]:=
```
i = 1; FindMinimum[f, {x, 0}, StepMonitor :> (Print["i=", i, "  x=", x]; i++)]
```

```
i=1  x=-0.5
```

```
i=2  x=-0.684912
```

```
i=3  x=-0.74196
```

```
i=4  x=-0.727817
```

```
i=5  x=-0.728439
```

```
i=6  x=-0.728447
```

```
i=7  x=-0.728447
```

FindMinimum::lstol :
The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances. ≫

Out[103]=
```
{-1.87577, {x → -0.728447}}
```

In[104]:=
```
i = 1; FindMinimum[f, {x, 0},
 EvaluationMonitor :> (Print["i=", i, "  x=", x]; i++)]
```

```
i=1  x=0.
i=2  x=-0.5
i=3  x=-1.66667
i=4  x=-0.684912
i=5  x=-0.74196
i=6  x=-0.727817
i=7  x=-0.728439
i=8  x=-0.728447
i=9  x=-0.728447
i=10  x=-0.728447
i=11  x=-0.728447
i=12  x=-0.728447
i=13  x=-0.728447
i=14  x=-0.728447
i=15  x=-0.728447
```

FindMinimum::lstol :
The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances. ≫

Out[104]=
```
{-1.87577, {x → -0.728447}}
```

■ Calkowanie

NIntegrate[funkcja, {zmienna, w_pocz, w_konc}]

In[105]:=
```
NIntegrate[Sin[x], {x, 0, 1}]
```

Out[105]=
```
0.459698
```

In[106]:=
```
Options[NIntegrate]
```

Out[106]=
```
{AccuracyGoal → ∞, Compiled → Automatic,
 EvaluationMonitor → None, Exclusions → None, MaxPoints → Automatic,
 MaxRecursion → Automatic, Method → Automatic, MinRecursion → 0,
 PrecisionGoal → Automatic, WorkingPrecision → MachinePrecision}
```

In[107]:=
```
NIntegrate[Sin[x]^2 Cos[y^2], {x, 0, Pi / 2}, {y, 0, Pi / 2}]
```

Out[107]=
```
0.666913
```

■ Rozwiazywanie rownan rozniczkowych

NDSolve[{rownanie,warunki}, funkcja,{zmienna, w_pocz, w_konc}]

In[109]:=
```
DSolve[y'[x] == Cos[x], y[x], x]
```

Out[109]=
```
{{y[x] → C[1] + Sin[x]}}
```

In[110]:=
```
r = DSolve[{y'[x] == Cos[x], y[0] == 1}, y[x], x]
```

Out[110]=
```
{{y[x] → 1 + Sin[x]}}
```

In[111]:=
```
f = y[x] /. r[[1]]
```

Out[111]=
```
1 + Sin[x]
```

In[112]:=
```
Table[f, {x, 0, 10}] // N
```

Out[112]=
```
{1., 1.84147, 1.9093, 1.14112, 0.243198,
 0.0410757, 0.720585, 1.65699, 1.98936, 1.41212, 0.455979}
```

In[113]:=
```
nr = NDSolve[{y'[x] == Cos[x], y[0] == 1}, y, {x, 0, 10}]
```

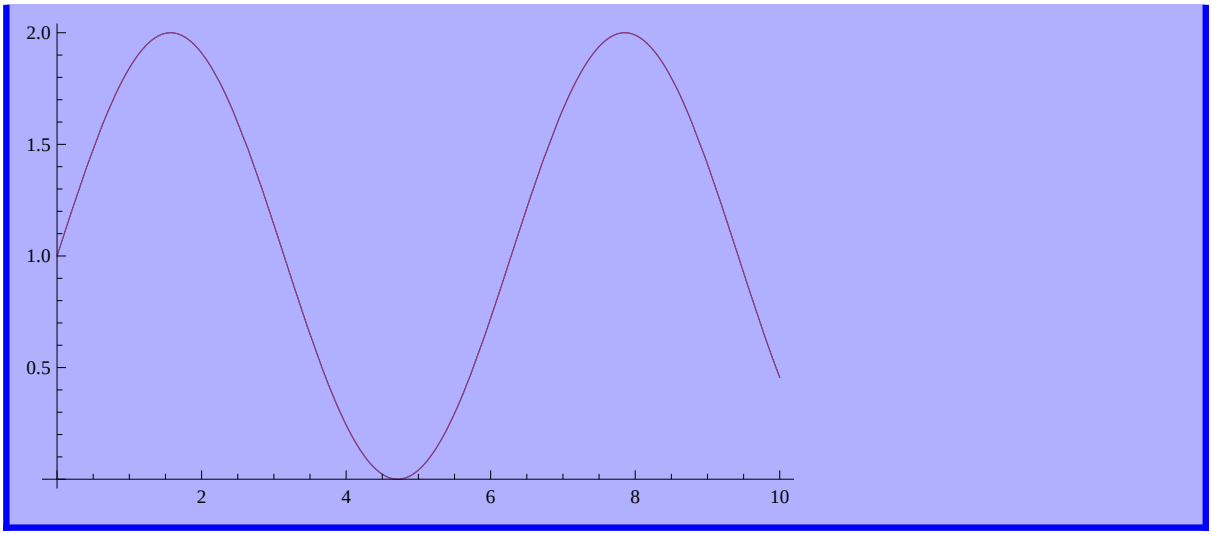
Out[113]=
```
{{y → InterpolatingFunction[{{0., 10.}}, <>]}}
```

In[114]:=
```
Table[y[x] /. nr[[1]], {x, 0, 10}]
```

Out[114]=
```
{1., 1.84147, 1.90929, 1.14112, 0.243198,
 0.0410775, 0.720585, 1.65698, 1.98935, 1.41212, 0.455979}
```

In[115]:=
```
Plot[{f, y[x] /. nr[[1]]}, {x, 0, 10}]
```

Out[115]=



In[116]:=
```
Options[NDSolve]
```

Out[116]=
```
{AccuracyGoal → Automatic, Compiled → Automatic,
 DependentVariables → Automatic, EvaluationMonitor → None,
 InterpolationOrder → Automatic, MaxStepFraction → 1/10,
 MaxSteps → 10 000, MaxStepSize → Automatic, Method → Automatic,
 NormFunction → Automatic, PrecisionGoal → Automatic,
 SolveDelayed → Automatic, StartingStepSize → Automatic,
 StepMonitor → None, WorkingPrecision → MachinePrecision}
```