
A *Mathematica* Tutorial -

1. Basic Operations

This Tutorial was created using Mathematica 6.0. Some of the cells described below can only be performed in version 6.0.

Note that Mathematica can be used as a word processor, a sophisticated calculator, a programming language that can be used for numerical calculations as well as symbolic calculations. It also includes a drawing tool that can be used to add whatever you like to your final document. You should do all of your homework for this course in Mathematica.

Please read the online [Practical Introduction to Mathematica](http://documents.wolfram.com/mathematica) at <http://documents.wolfram.com/mathematica>

It is important that you also do the "First Five Minutes with Mathematica" tour provided on the Mathematica 6 Startup Palette.

A complete online manual can be found by following the menu item "Help" → "Documentation Center". This provides text descriptions for the various features as well as examples.

The full manual for Mathematica can be downloaded from the Documentation web site as a pdf file.

Notebooks

Mathematica documents are called "notebooks". They can contain text, commands, and graphics.

All entries appear in "cells" which are delineated by brackets at the right side of the page.

Various pre-set "styles" are available for your notebooks under the menu "Format" → "Stylesheet". Try them out to see which you prefer.

The format of the text is controlled by the user with various styles appearing under the menu item "Format" → "Style". In the section headings below the background color is "grey cell box".

Calculations and variables

Start a new cell by clicking the cursor in an open space and type the text or calculation that you want to perform. Start by typing the following followed by hitting the "enter" key to evaluate the expression and obtain the answer in an output cell :

```
2 + 2
```

```
4
```

Multiplication can be indicated by a space, e.g. $2 * 4$ can be written with the asterisk, or with $2 \cdot 4$

```
2 \cdot 4
```

```
8
```

Exponentiation is indicated by the upward carrot sign :

```
2 ^ 4
```

```
16
```

The most recent result of a calculation is symbolized by "%" :

```
2 %
```

```
32
```

A variable can be assigned a value with the equals sign :

```
y = 2 ^ 6 + 32
```

```
96
```

You can obtain the value of a variable at anytime by using it in an expression (or simply typing it followed by a return) .

```
y
```

```
96
```

The value of any variable can be removed with a period following an equals sign :

```
y = .
```

```
y
```

```
y
```

Important :

Note that many users of Mathematica can become confused by its performance since it does not normally function like a C or Fortran program by running from top to bottom. Keep in mind that it is a notebook, and once a variable or function is defined, it keeps that value even if you move back to a line above where it was most recently defined and perform another evaluation.

You can run a Mathematica notebook as a program from top to bottom by evaluating all the initialization cells (Kernel → Evaluation → Evaluate Initialization). Select each cell that you want to specify as an Initialization Cell and go to Cell → Cell Properties → Initialization Cell. The cell below is an initialization cell since it has an I at the top of the cell bracket. The semicolon at the end of each line prevents that equality from being repeated as an output line.

```
I y = 2 Pi;  
x = 360;  
z = 2 ^ 24;
```

Functions defined in *Mathematica*

Functions are indicated by keywords that are capitalized. Sin is a reserved word indicating the Sine function. The argument of a function is indicated in square brackets following the keyword. Pi is a reserved constant. Note that the first letter of reserved function or constant is always capitalized.

```
Sin[2 Pi]
```

```
0
```

The argument of trig functions is in radians by default.

```
Sin[Pi / 2]
```

```
1
```

You can switch to degrees by writing "Degree" after the argument (Note the unit Degree is always singular in Mathematica) :

```
Sin[90 Degree]
```

```
1
```

Numerical evaluation can be forced with the N[] function :

```
Pi
```

```
 $\pi$ 
```

```
N[Pi]
```

```
3.14159
```

The level of precision can be specified as a second element in the N function.

```
N[Pi, 100]
```

```
3.141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117068
```

Functions defined by the user

You can define our own functions. Note that the definition specifies the variable on the left side in brackets with the underline following it. This is to be substituted into the expression on the right. The function can be used anywhere it is needed with the value for the variable explicitly indicated in brackets.

```
ANewFunc[x_] := x^2 + 2 x + 3
```

```
y = ANewFunc[20]
```

```
443
```

The definition of any function can be removed with an equals sign followed by a period :

```
ANewFunc[x_] = .
```

```
y = ANewFunc[20]
```

```
ANewFunc[20]
```

Logarithms

The natural log of a number is obtained with the `Log[]` function :

```
Log[10.0]
```

```
2.30259
```

The log of a number to the base 10 is obtained with the `Log[10, x]` :

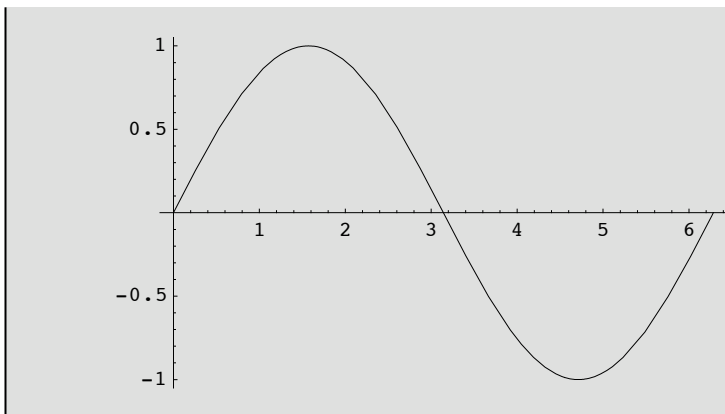
```
Log[10, 2.0]
```

```
0.30103
```

Plotting

Use the `Plot` function to plot the a function,
with the limits for the plot indicated in curly brakets :

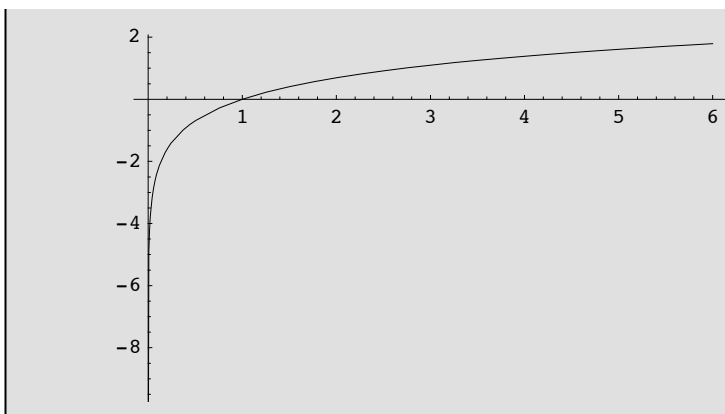
```
Plot[Sin[x], {x, 0, 2 Pi}]
```



- Graphics -

See the online documentation and manual for more complete control of axes and labels.

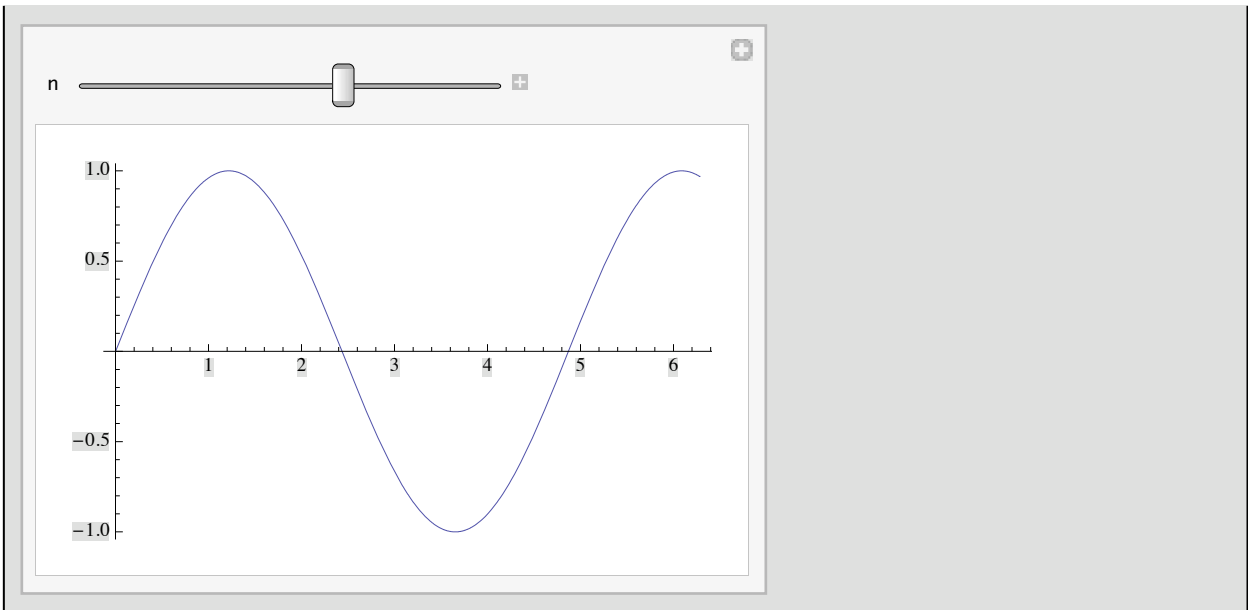
```
Plot[Log[x], {x, 0, 6}]
```



- Graphics -

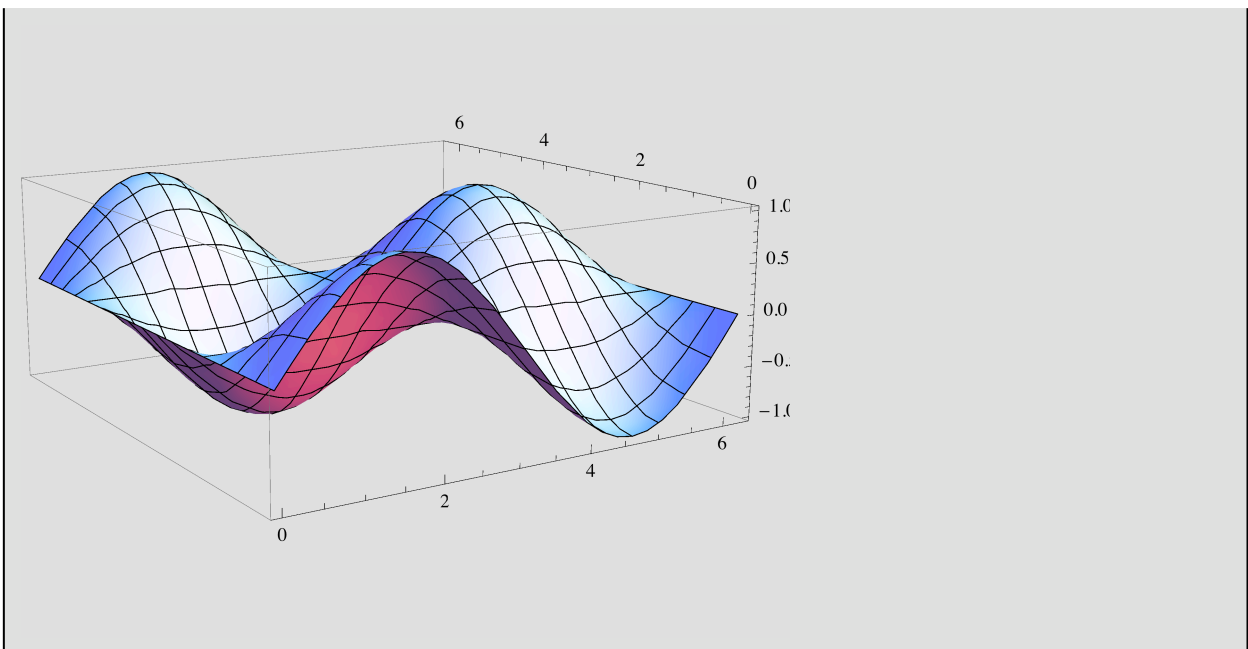
The `Manipulate` command permits the addition of a slide bar to a plot to allow real time adjustment of parameters. In the example below, the parameter is adjustable from 0 to 2 with the initial value set to 1.

```
Manipulate[Plot[Sin[n x], {x, 0, 2 Pi}], {{n, 1}, 0, 2}]
```



3 D plots are obtained with the Plot3D function :

```
Plot3D[Sin[x] Cos[y], {x, 0, 2 Pi}, {y, 0, 2 Pi}]
```



Note : The plot can be selected and rotated in real time to change the view.
The graphics output can be selected, copied, and pasted into a report or Powerpoint presentation.

Calculus

Derivatives can be calculated with the `D[,]` function with the first element indicating the function to be evaluated, and the second element indicating the variable the derivative is to be calculated with respect to. Use the Basic Math Palettes for a more traditional input format (as shown here) :

$$\partial_x x^2$$

$$2 x$$

$$\partial_x \text{Log}[x]$$

$$\frac{1}{x}$$

Integration is calculated with the `Integrate[,]` function with the function in the first element, and the variable to be integrated in the second element. The Basic Math Palette provides a traditional input format :

$$\int x^2 dx$$

$$\frac{x^3}{3}$$

Note that these are indefinite integrals without limits. Definite integration with limits is also possible :

$$\int_0^2 x^2 dx$$

$$\frac{8}{3}$$

Data input and plotting

The `Directory[]` command returns the current Directory :

```
Directory[]
```

```
/Users/johnshriver
```

You can also redefine the current directory :

```
SetDirectory["/Users/johnshriver/Desktop"]
```

```
/Users/johnshriver/Desktop
```

Set up a matrix of x and y values using the "Insert" → "Table, Matrix" menu option, or the Basic Math Palette :

```
Mydata = 
$$\begin{pmatrix} 0 & 0 \\ 1 & 2 \\ 2 & 3.5 \\ 3 & 4 \\ 4 & 4.2 \\ 5 & 4.3 \\ 6 & 4.4 \\ 7 & 4.4 \end{pmatrix}$$

```

```
{{0, 0}, {1, 2}, {2, 3.5}, {3, 4}, {4, 4.2}, {5, 4.3}, {6, 4.4}, {7, 4.4}}
```

If the data to be input is already in a file (e.g. a WORD text file, or an Excel file, the data can be imported using the Import command :

```
Import["testdata.dat", "Table"]
```

```
{{1, 1.1}, {2, 2.}, {3, 3.2}, {4, 3.9}, {5, 4.9}}
```

The above command could be modified to give the data set a name :

```
TestData = Import["testdata.dat", "Table"]
```

```
{{1, 1.1}, {2, 2.}, {3, 3.2}, {4, 3.9}, {5, 4.9}}
```

```
TableForm[TestData]
```

```
1  1.1
2  2.
3  3.2
4  3.9
5  4.9
```

Data can also be Exported as a text file :

```
Export["output_test.txt", Mydata]
```

```
output_test.txt
```

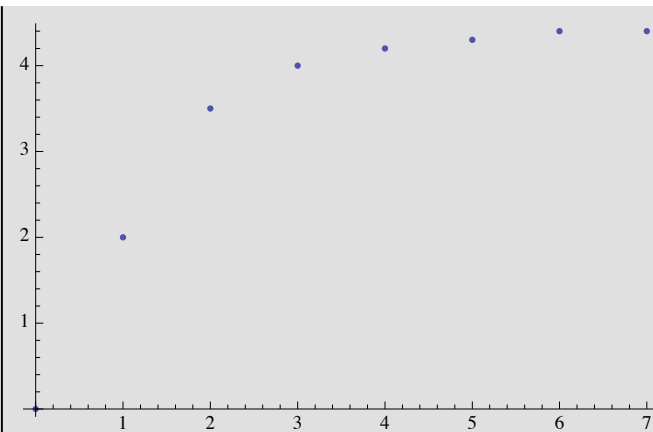
Export into an Excel format :

```
Export["output.xls", Mydata]
```

```
output.xls
```

Plot data using the ListPlot Command instead of Plot :

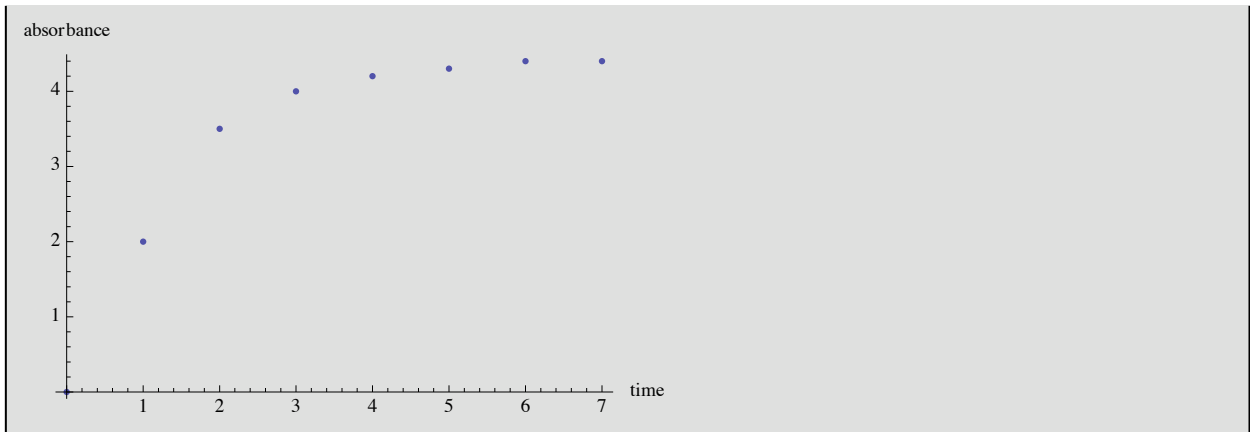
```
ListPlot[Mydata]
```



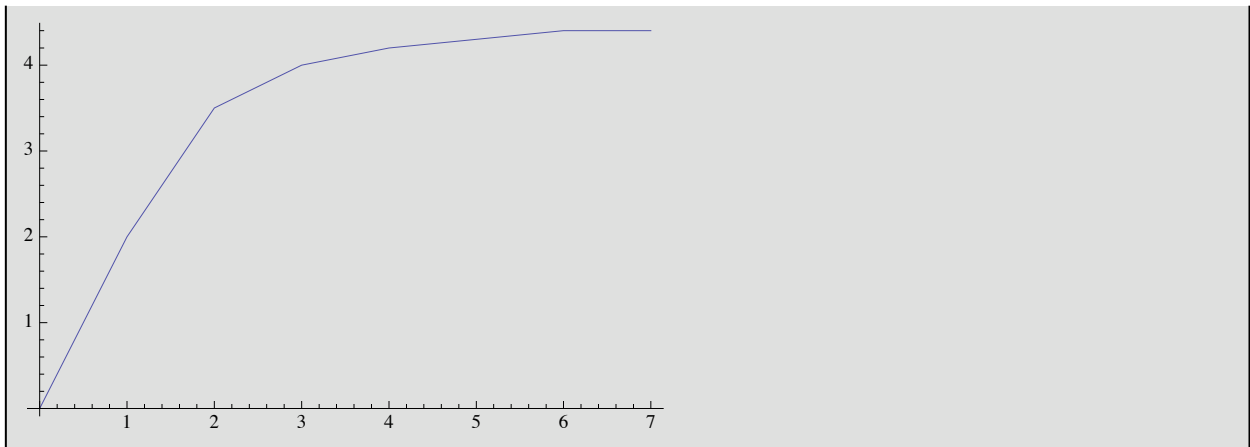
See Help for ListPlot options. Forexample,
the point size can be controlled with the Prolog → AbsolutePointSize[] function.
A frame on all four sides can be added with Frame → True.

For later reference, the plot can be given a name :

```
dataplot = ListPlot[Mydata, Prolog -> AbsolutePointSize[5], AxesLabel -> {time, absorbance}]
```



```
ListPlot[Mydata, PlotJoined -> True]
```



Data analysis - Fitting data

A simple fit of the data using a linear least square routine could be obtained with the Fit function.

```
Fit[Mydata, {1, x}, x]
```

```
1.45833 + 0.540476 x
```

Of course a linear fit of this data is not appropriate here. To obtain a nonlinear fit we need to load the Statistics Nonlinear package.

```
Needs["NonlinearRegression`"];
```

The function `NonlinearFit` requires the following format :

```
NonlinearFit[data, model, {parameters}, {variables}]
```

The model is the mathematical function which is assumed to describe the data. The list of variables and parameters follow. Typically, the variables list will only contain a single variable, while the parameters to be fit may contain two or more items.

A simple example would be a fit of the above data with a linear equation (of course a linear fit would normally be accomplished the linear least squares program `Fit[]`, but it works here as an introductory example).

```
NonlinearRegress[Mydata, a + b x, {a, b}, x]
```

```
{BestFitParameters -> {a -> 1.45833, b -> 0.540476},
```

	Estimate	Asymptotic SE	CI
ParameterCITable -> a	1.45833	0.592264	{0.00911456, 2.90755}
b	0.540476	0.141578	{0.194047, 0.886906}

```
EstimatedVariance -> 0.841865,
```

	DF	SumOfSq	MeanSq
Model	2	102.049	51.0244
ANOVATable -> Error	6	5.05119	0.841865,
Uncorrected Total	8	107.1	
Corrected Total	7	17.32	

```
AsymptoticCorrelationMatrix -> ( 1.      -0.83666 )
  -0.83666  1. ) ,
```

	Curvature
FitCurvatureTable -> Max Intrinsic	0
Max Parameter-Effects	0
95. % Confidence Region	0.440942

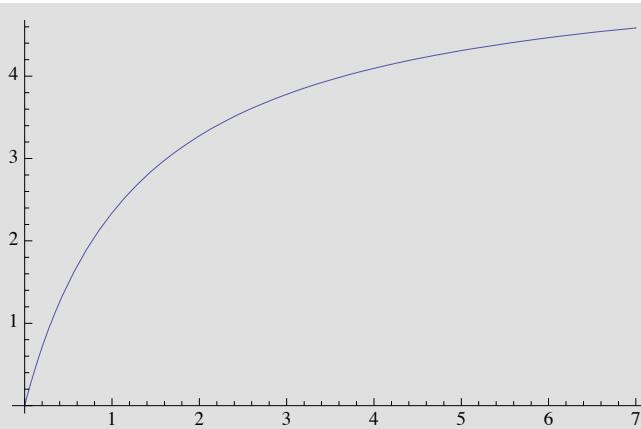
A more appropriate function is normally obtained from an understanding of the experiment (the user usually knows what function to fit the data to based on the experimental situation, alternatively, he can fit the data to an arbitrary function, e.g. a polynomial).

```
NonlinearRegress[Mydata, a x / (x + b), {a, b}, x]
```

```
{BestFitParameters → {a → 5.4583, b → 1.33173},
ParameterCITable → a | Estimate Asymptotic SE CI
                    b | 5.4583 0.291469 {4.7451, 6.1715}
                    | 1.33173 0.261235 {0.692517, 1.97095}
EstimatedVariance → 0.0440968,
ANOVA Table →
      Model          DF      SumOfSq      MeanSq
      Error          6      0.264581    0.0440968,
      Uncorrected Total 8      107.1
      Corrected Total  7      17.32
AsymptoticCorrelationMatrix → ( 1.      0.924797 )
                               ( 0.924797 1. )
FitCurvatureTable →
      Max Intrinsic          0.0735356
      Max Parameter-Effects 0.248355
      95. % Confidence Region 0.440942
```

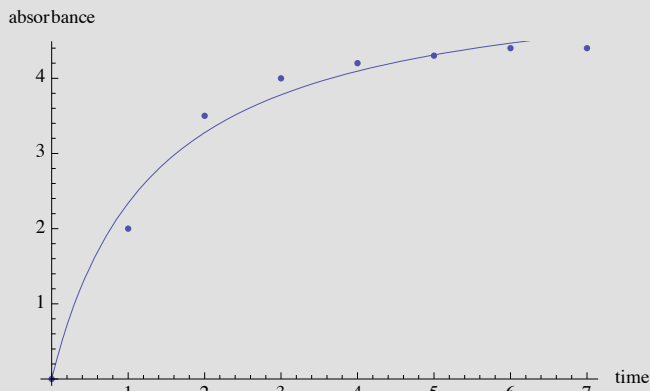
A plot of the fitted function (with the plot given a name "fit" here) :

```
fit = Plot[5.45833 x / (1.33177 + x), {x, 0, 7}]
```



Overlay the data and the fitted function with the Show function :

```
Show[dataplot, fit]
```



```
NonlinearRegress[Mydata, a x / (x + b), {a, b}, x, ShowProgress -> True,
  RegressionReport -> {BestFitParameters, ParameterCITable}]
```

```
Iteration:1 ChiSquared:4.96156 Parameters:{5.46347, 2.69759}
Iteration:2 ChiSquared:2.16993 Parameters:{5.20513, 0.628268}
Iteration:3 ChiSquared:0.311908 Parameters:{5.36803, 1.16713}
Iteration:4 ChiSquared:0.264682 Parameters:{5.4705, 1.33979}
Iteration:5 ChiSquared:0.264581 Parameters:{5.45724, 1.33071}
Iteration:6 ChiSquared:0.264581 Parameters:{5.45843, 1.33186}
Iteration:7 ChiSquared:0.264581 Parameters:{5.45828, 1.33172}
Iteration:8 ChiSquared:0.264581 Parameters:{5.4583, 1.33174}
Iteration:9 ChiSquared:0.264581 Parameters:{5.4583, 1.33173}
Iteration:10 ChiSquared:0.264581 Parameters:{5.4583, 1.33173}
Iteration:11 ChiSquared:0.264581 Parameters:{5.4583, 1.33173}
```

```
{BestFitParameters -> {a -> 5.4583, b -> 1.33173},
ParameterCITable ->


|   | Estimate | Asymptotic SE | CI                  |
|---|----------|---------------|---------------------|
| a | 5.4583   | 0.291469      | {4.7451, 6.1715}    |
| b | 1.33173  | 0.261235      | {0.692517, 1.97095} |


}
```