**[DOING PHYSICS WITH MATLAB](http://www.physics.usyd.edu.au/teach_res/mp/mphome.htm)**

**DATA ANALYSIS**

**WEIGHTED FIT**

Ian Cooper

School of Physics, University of Sydney

ian.cooper@sydney.edu.au

**MATLAB SCRIPTS**

[Goto the directory containing the m-scripts and data files.](http://www.physics.usyd.edu.au/teach_res/mp/mscripts)

The Matlab scripts that are used to fit an equation to a set of experimental data:

weighted.m m-script used to fit a straight line to set of experimental data

which calls the following functions

fitFunction.m Function used to evaluate the fitted function

partDev.m Function to evaluate partial derivatives of the fitted function

chi2test.m Function to evaluate the *χ*2 value

Sample data files: data1A.mat data1B.mat data1C.mat

wData1.mat wData2.mat wData3.mat wData4.mat

**CURVE FITTING**

**LEAST SQUARES - UNCERTAINITES IN THE DATA**

In many experiments, the functional relationship between two variables *x* and *y* is investigated by measuring a set of *n* values of (*xi*, *yi*) and their uncertainty σ*xi*, σ*yi*. The functional relationship between *x* and *y* can be written as

(1) *yf* = *f*(*a*1, *a*2, … , *a*m; *x*) = *f*(**a**; *x*)

The goal is to find the unknown coefficients **a** = {*a*1, *a*2, … , *a*m} to fit the function *f*(**a**; *x*) to the set of *n* measurements. For a statistical analysis, it is difficult to consider simultaneously the uncertainties in both the *x* and *y* values. In this treatment, only the uncertainties **σ*yi*** in the *y* measurements will be considered.

For the method of least squares, to find the coefficients **a**, the best estimates are those that minimizes the χ2 value,given by equation (2)

(2) 

This is simply the sum of the squared deviations of the measurements from the fitted function *f*(**a**; *x*) weighted by the uncertainties σ*yi* in the *y* values.

The default m-script **weighted.m** can be used to trial any one of nine different functions to fit a set of experimental data.

1. *y* = *a*1 *x* + *a*2 (linear)
2. *y* = *a*1 *x* (*y* proportional to *x*)
3. *y* = *a*1 *x*2 + *a*2 *x* + *a*3 (parabolic)
4. *y* = *a*1 *x*2 (parabolic)
5. *y* = *a*1 *x*3 + *a*2 *x*2 + *a*3*x* + *a*4 (cubic)
6. *y* = *a*1 *x*a2 (power)
7. *y* = *a*1 exp(-*a*2 *x*) (exponential: *x* ≠ 0, *y* ≠ 0)
8. *y* = *a*1[1 - exp(-*a*2 *x*)] (exponential: y ≠ 0)
9. *y* = *a*1 *x*4 + *a*2 *x*3 + *a*3*x*2 + *a*4 *x* + *a*5 (polynomial 4th order)

This m-script calls two extrinsic functions: **fitFunction** to evaluate the fitted function and **partDev** to calculate the partial derivative of the function with respect to the coefficients, . Also there is call to the m-script **chi2test.m** to give a measure the “goodness of the fit”. An iterative procedure is used based upon the method of Marquardt where the minimum of χ2 is found by adjusting the value of the coefficients through a damping factor *u*.

Details are given in the following sections so that you can modify the m-script **weighted.m** to add your own functions to fit a set of measurements. Also the m-script **chi2test.m** can be run as independent program.

**CHI-SQUARED DISTRIBUTION**

The **chi-squared distribution χ2** (χ2 is a single entity and is not equal to χ×χ) and is very useful for testing the **goodness-of-of fit** of a theoretical equation to a set of measurements. For a set of *n* independent random variables *xi* that have a Gaussian distribution with theoretical means *μi* and standard deviations *σi*, the chi-squared distribution χ2 defined as

(3) 

χ2 is also a random variable because it depends upon the random variables *xi* and μ*i* and follows the distribution

(4) 

where Γ( ) is the **gamma function** and *ν* is the number of **degrees of freedom** and is the sole parameter related to the number of independent variables in the sum used to describe the distribution. The mean of the distribution is *μ* = *ν* and the variance is *σ*= 2*ν*. The **reduced chi-squared** value is defined as

(5) 

**This distribution can be used to test a hypothesis that a theoretical equation fits a set of measurements**.

If an improbable chi-squared value is obtained, one must question the validity of the fitted equation. Basically, we have set up a hypothesis that our measurements can be described by some analytical function *f*(**a**; *x*). We test the hypothesis by the value of χ2. χ2 is a measure of the total agreement between our measurements and the hypothesis. It can be assumed that the minimum value of χ2 is distributed according to the χ2 distribution with *ν* = (*n*-*m*) degrees of freedom (*n* data points and *m* coefficients in the fitting function).

The **reduced χ2** value{χ2reduced = χ2/(*n*-*m*)} is quoted as a measure of the **goodness-of-fit**

χ2reduced ~ 1 ⇒ hypothesis is acceptable

χ2reduced << 1 ⇒ the fit is much better than expected given the size of the measurement uncertainties. The hypothesis is acceptable, but the uncertainties *σy* may have been overestimated.

χ2reduced >> 1 ⇒ hypothesis may not be acceptable

The extrinsic function **chi2test.m** can be used to display the distribution for a given degree of freedom ν and gives the **probability** of a chi-squared value exceeding a given chi-squared value.



This function **ch12test.m** can be run independent of the m-script **weighted.m**

For example, chi2test(6, 12) → prob = 6.2 %. This would imply that the hypothesis should be **rejected** because there is only a relatively small probability that χ2 = 12 with ν = 6 degrees of freedom would occur by chance. The χ2 distribution given by equation (4) is shown in Fig. 1.



Fig. 1 Chi-squared distribution with parameters: 6 degrees of freedom, the χ2 value = 12 and the probability = 6.17% that the χ2 would be exceeded by chance.

A much higher value for the probability indicates it is more likely that the hypothesis is acceptable.

**USING THE M-SCRIPT weighted.m**

The measurements must be entered into a matrix called **wData** of dimension (*n*×4) where *n* is the number of data points. The analysis only uses the uncertainties *σy* associated with the *y* measurements. The uncertainties *σx*are only included to show any error bars when the measurements are plotted.

|  |  |
| --- | --- |
| **Measurements** | **Matrix** |
| *x* | x = wData(:, 1) |
| *y* | y = wData(:, 2) |
| *σx* | dx = wData(:, 3) |
| *σy* | dy = wData(:, 4) |

It is **necessary** to clear all variables from the Workspace using the Matlab command clear all.

The data (*x*, *y*,*σx*, *σy*) can be entered into the matrix wData1 and saved to a file from the Command Window using the Matlab command **save**. To use this data at any time use the Matlab command **load** to retrieve the variable. Then let wData = wData1 before running the m-script for **weighted.m**.

Getting started in the Command Window:

clear all

close all

clc

wData1 = zeros(9,4) for 9 data points

Goto to the Workspace for the variable wData1 and enter data directly into the cells of the matrix displayed.

Save the data:

save wData1 wData1 (file name variable)

Load the data:

load wData1

Assign the data to wData:

wData = wData1

Run the program fitting program:

weighted

You then will be prompted to enter:

* The minimum value for the fitted function.
* The maximum value for the fitted function.
* The title for the plot.
* The *X* axis label.
* The *Y* axis label.
* The equation to be fitted to the data by entering a number from 1 to 9

1: y = a1 \* x + a2

2: y = a1 \* x

3: y = a1 \* x^2 + a2 \* x + a3

4: y = a1 \* x^2

5: y = a1 \* x^3 + a2 \* x^2 + a3 \* x + a4

6: y = a1 \* x^a2

7: y = a1 \* exp(- a2 \* x) (x ≠0 y ≠ 0)

8: y = a1 \* (1 - exp(-a2 \* x)) (y ≠ 0)

9: y = a1 \* x^4 + a2 \* x^3 + a3 \* x^2 + a4 \* x +a5

After the m-script has been executed, the following information is displayed in the Command Window:

* The equation type.
* Number of measurements.
* The degrees of freedom (*n* – *m*) where *n* is the number of measurements and *m* is the number of coefficients (*a*1, *a*2, …, *a*m ).
* Measure s for the “good-of-fit” - χ2 value, reduced χ2 value, and a probability factor and a description of the fitted equation to the data.
* Coefficients in the array **a** {*a*1, *a*2, ... , *a*m).
* The uncertainties in the coefficients in the array **sigma**.

The results are also displayed in two Figure Windows:

* The *XY* plot of the measurements and the fitted equation.
* The χ2  distribution indicating the “good-of-fit”

**Example 1 (Linear Fit y = a1 \* x + a2)**

We will consider the data concerning the extension *x* of a spring caused by a load *F*. For an ideal spring, the relationship between the applied force *F* and the extension *e* is given by **Hooke’s Law**

*F* = *k e* (*F* directly proportional to *x*)

where *k* is known as the spring constant. A graph of *F* vs *e* corresponds to a straight line through the origin (0,0) and the slope of the line gives the value of the spring constant *k*.

Different sets of data will be considered to illustrate the ways in which the weighted curve fitting program can be used and how we can test the hypotheses that a straight line fits the data.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| wData(:,1) | wData(:,2) | wData(:,3) | wData(:,4) | wData(:,3) | wData(:,4) | wData(:,3) | wData(:,4) |
| *x*: *e* (mm) | *y*: *F* (N) | *σx* (mm) | σ*y* (N) | *σx* (mm) | σ*y* (N) | *σx* (mm) | σ*y* (N) |
| 0 | 0 | 0 | 0 | 2 | 0.05 | 0 | 0 |
| 20 | 0.50 | 0 | 0 | 2 | 0.05 | 1 | 0.03 |
| 55 | 1.00 | 0 | 0 | 2 | 0.05 | 3 | 0.05 |
| 78 | 1.50 | 0 | 0 | 2 | 0.05 | 4 | 0.07 |
| 98 | 2.00 | 0 | 0 | 2 | 0.05 | 5 | 0.10 |
| 130 | 2.50 | 0 | 0 | 2 | 0.05 | 6 | 0.13 |
| 154 | 3.00 | 0 | 0 | 2 | 0.05 | 8 | 0.15 |
| 173 | 3.50 | 0 | 0 | 2 | 0.05 | 9 | 0.17 |
| 205 | 4.00 | 0 | 0 | 2 | 0.05 | 10 | 0.20 |
| data saved as | | data1A | | data1B | | data1C | |
| chi-squared | | 0.0384772 | | 15.3909 | | 13.8177 | |
| reduced chi-squared | | 0.00549675 | | 2.1987 | | 1.97395 | |
| probability % | | 100 | | 3.11 | | 5.43 | |
| Fit | | Maybe too good ? | | Acceptable | | Acceptable | |
| slope: coefficient *a*1 | | 0.0196 | | 0.0196 | | 0.0195 | |
| uncertainty in *a*1 | | 0.0051 | | 0.0003 | | 0.0005 | |
| intercept: coefficient *a*2 | | 0.0147 | | 0.0147 | | 0.0203 | |
| uncertainty in *a*2 | | 0.6120 | | 0.0306 | | 0.0346 | |
| spring constant *k*  (N.m-1) | | 20 ± 5 | | 19.6 ± 0.3 | | 19.5 ± 0.5 | |
| intercept (N) | | 0.01 ± 0.6 | | 0.01 ± 0.03 | | 0.02 ± 0.03 | |

Table 1. Fitting the spring data to the linear function y = a1 \* x + a2 for three different sets of uncertainties in *x* and *y*.

For comparison, using the non-weighted linear fitting program **linear\_fit.m** for the *x* and *y* data gave the following results:

Fit: *y* = *m x* + *b*

n = 9

slope m = 0.01957 Em = 0.0003751 ⇒ *k* = (19.57 ± 0.04) N.m-1

intercept *b* = 0.0147 Eb = 0.04537 ⇒ intercept *b* = (0.01 ± 0.04)

correlation r = 0.9987

By examining the results shown in Table 1, the weighted fit over-estimates the uncertainties in the coefficients when all the uncertainties in the data are zero. For the case when all the uncertainties in the data are zero, it is better to use the non-weighted linear fitting method.

Figures 2 and 3 shows the graphical output for the uncertainties given in columns 7 and 8 of Table 1.



Fig. 2. Linear fit to the data y = a1 \* x + a2. The slope of the line is (19.5 ± 0.5 N) and the intercept is (0.2 ± 0.3).



Fig. 3. Ch-squared distribution. The χ2 ~ 1, so the fit is acceptable although the probability of the weighted least squares is quite small.

**Example 2 (Power relationship** **)**

The data below is used to fit a power relation . Data stored as wData2.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*: *m* (kg) | 0.02 | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 | 0.50 |
| *y*: *T* (s) | 0.20 | 0.31 | 0.46 | 0.62 | 0.71 | 0.72 | 0.76 | 0.84 | 0.89 | 0.90 |
| *σx* (kg) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| σ*y* (s) | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.10 | 0.10 |

The results of the weighted least squares fit are:

6: y = a1 \* x^a2 power

No. measurements = 1e+01

Degree of freedom = 8

chi2 = 4.27503

Reduced chi2 = 0.534379

Probability = 83.1

\*\*\* Acceptable Fit \*\*\*

Coefficients a1, a2, ... , am

1.3498

0.4540

Uncertainties in coefficient

0.0963

0.0458

Hence, we can conclude the power fit is an acceptable fit to the data with coefficients: *a*1 = (1.3 ± 0.1) s.kg-1 *a*2 = (0.45 ± 0.05)

The graphical outputs are shown in figures 4 and 5.

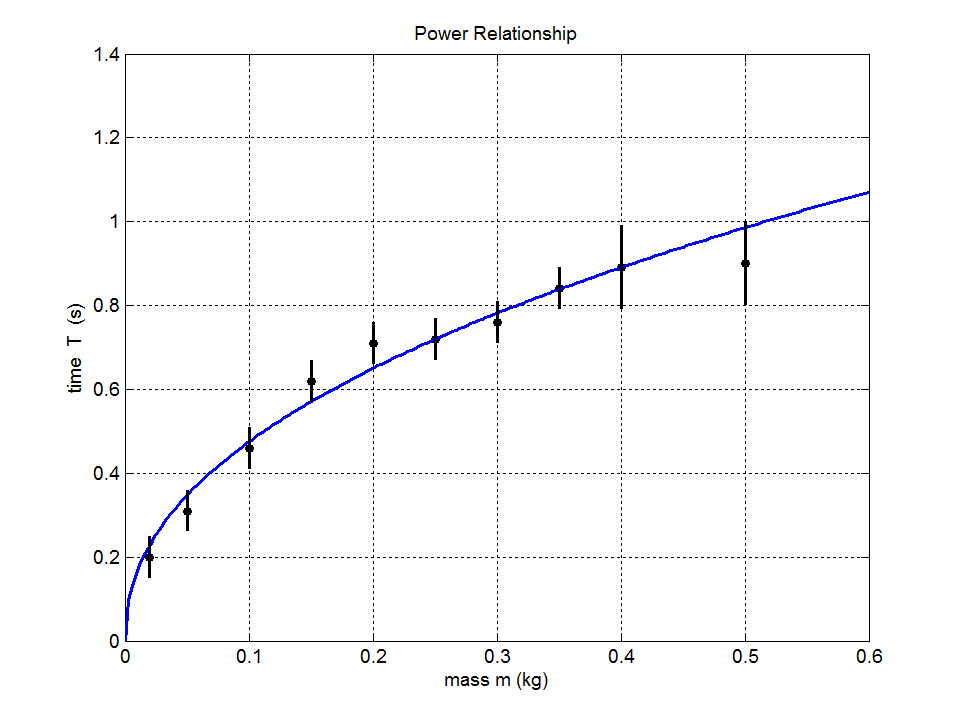
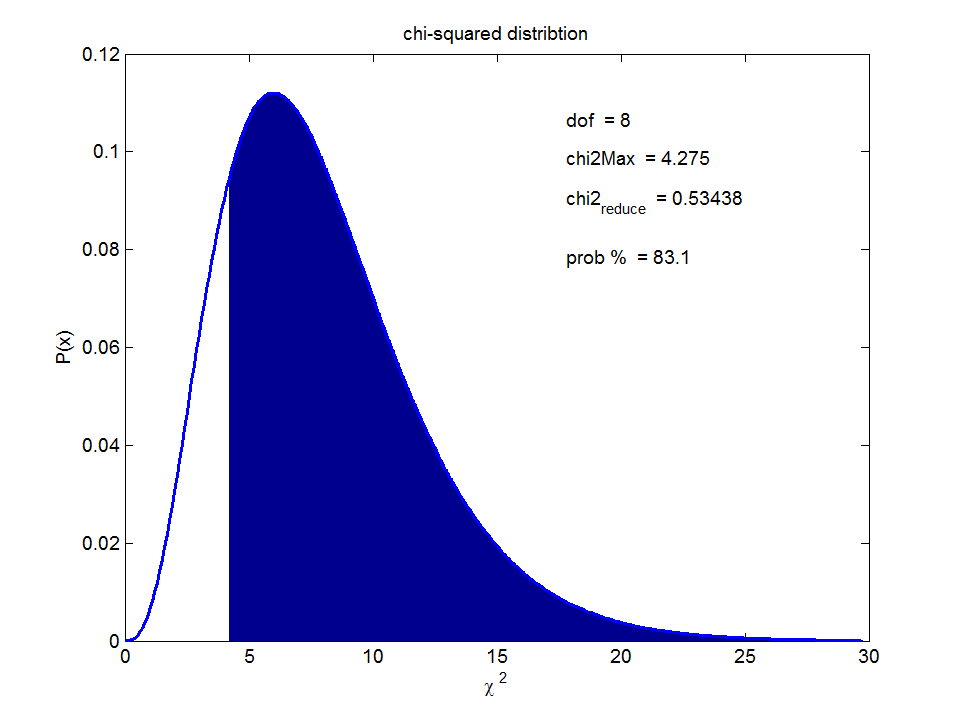


Fig. 4. Power fit  to the experimental data.

Fig.5. Plot of the chi-squared distribution showing the probability of χ2 value being exceeded.

**Example 3 (exponential decay** **)**

The exponential relation  is used to fit to the experimental data saved as wData3.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*: *t* (s) | 0 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 |
| *y*: *A* (counts) | 20 | 12.5 | 8.0 | 5.0 | 3.5 | 2.5 | 1.5 | 1.0 | 0.5 |
| *σx* (s) | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| σ*y* (counts) | 1.0 | 1.0 | 1.0 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |

The results of the weighted least squares fit are:

7: y = a1 \* exp(- a2 \* x) exponential decay

No. measurements = 9

Degree of freedom = 7

chi2 = 1.06625

Reduced chi2 = 0.152321

Probability = 99.3

? Fit may be too good ?

Coefficients a1, a2, ... , am

20.0000

0.0444

Uncertainties in coefficient

0.8857

0.0023

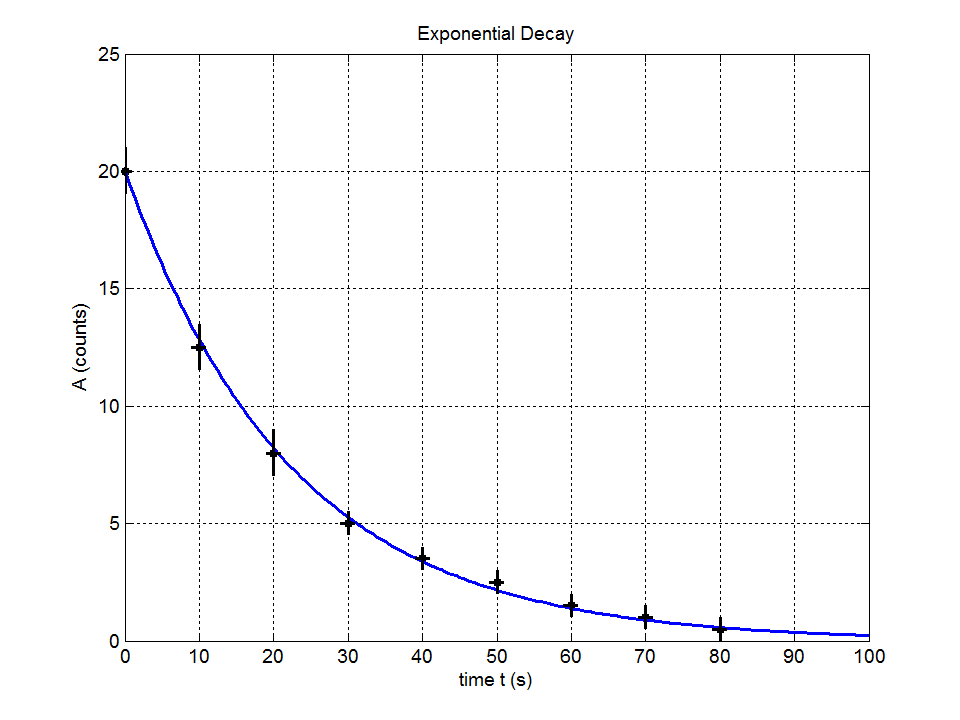


Fig. 6. Plot of data and exponential decay fit for data in Example 3.

**Example 4 Interpolation**

The **weighted-fit** m-script can be used for **interpolation**. For example, the viscosity η of water is a function of temperature *T* and tables give the viscosity at only fixed temperatures. By fitting a polynomial to the data, one can estimate the viscosity at a temperature between the fixed values.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*: *T* (°C) | 0 | 10 | 20 | 30 | 40 | 50 | 60 | 80 | 100 |
| *y*: *η* (mPa.s) | 1.783 | 1.302 | 1.002 | 0.800 | 0.651 | 0.548 | 0.469 | 0.354 | 0.281 |
| *σx* (oC) | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| σ*y* (mPa.s) | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 |

Data stored as wData4. In the Command Window type format shorte to display the results in scientific notation

|  |  |  |
| --- | --- | --- |
| **Type of fit** | **3rd order polynomial (5)** | **4th order polynomial (9)** |
| chi-squared | 168.672 | 13.5519 |
| reduced chi-squared | 33.7343 | 3.38797 |
| probability % | 0 | 0.866 |
| Fit | may not be acceptable | may not be acceptable |
| coefficient *a*1 | -(2.68 ± 0.07)×10-6 | (3.4 ± 0.3)×10-8 |
| coefficient *a*2 | (6.0 ± 0.1)×10-4 | -(9.5 ± 0.5)×10-6 |
| coefficient *a*3 | -(4.76 ± 0.04)×10-2 | (1.01 ± 0.03)×10-3 |
| coefficient *a*4 | (1.756 ± 0.004) | -(5.56 ± 0.08)×10-2 |
| coefficient *a*5 | --- | (1.778 ± 0.005) |
| Table: T at 20 oC | 1.002 | 1.002 |
| Predicted T at 20 oC | 1.021 | 0.996 |
| Predicted T at 24 oC | 0.9200 | 0.9055 |

The 4th order polynomial has a much lower reduced χ2 value and gives a better fit to the data than the 3rd order polynomial.

|  |  |
| --- | --- |
|  |  |

Fig. 7. 3rd and 4th order polynomial fit to viscosity data.

**THE METHOD OF LEAST SQUARES**

Step 1: Assigning the weights for finding uncertainties

Weights **w** are assigned to the uncertainties **dy** in the y measurements

**w = 1/dy** if dy*k* = 0 then w*k* = 1 for any *k*

An adjustable parameter u known as the damping factor is initially set to 0.001 so that the coefficients **a** can be adjusted to minimize the χ2 value by simply adjusting the value of u.

Step 2: Set the starting values for the coefficients **a**

To use the least squares method, we have to estimate starting values for the coefficients **a**. If the equation can be made linear in some way, then we can solve *n* simultaneous equations to find the unknown values of **a**. For example,

EqType = 5

% f = a1 \* x^3 + a2 \* x^2 + a3 \* x + a4 cubic polynomial

xx(:,1) = x.^3;

xx(:,2) = x.^2;

xx(:,3) = x;

a = xx\y;

If this can’t be done, a simpler method is used to set the coefficients or **a** = 1.

Step 3: Minimize χ2 value

The counters for the number of data points *n* and number of coefficients *m* are

*i* = 1, 2, …, *n*

*k* = 1, 2, … , *m*

*j* = 1, 2, … , *m*

The weighted difference matrix **D** is

 ⇒ **D** = w**’** \* (y – f)

The value of χ2 is then

χ2 = **D’** \* **D**

where **’** gives the transpose of a matrix.

We need to adjust the coefficients by an iterative method until we find the true minimum of χ2. For step *L* in the iterative procedure

**a**(*L*+1) = **a**(*L*) + **da**

and our desired goal is that χ2{**a**(*L*+1)} < χ2{**a**(*L*)}.

The minimum of χ2(**a**) is given by the condition



For small variations of the coefficients, the value of χ2{**a**(*L*+1)} may be expanded in terms of a Taylor’s series around χ2{**a**(*L*)} and if the expansion is truncated after the second term, we can use the approximation



This can be written in matrix form as

**B** = **CUR** \* **da**

where  and 

where **da** is the matrix for the increments in the coefficients, **CUR** is called the **curvature matrix** as it expresses the curvature of χ2(**a**) with respect to **a**.

The **B** matrix, after performing the partial differentiation can be written as



We need to calculate the partial derivatives of the fitted function with respect to the coefficients **a**. To make the program more general, the weighted partial derivates **pdf** are calculated numerically by the function **part\_der** using the difference approximation to the derivative



The matrix **B** written in matrix form is

**B = pdf’ \* D**

The curvature matrix **CUR** can be approximated by



**CUR = pdf’ \* pdf**

The elements of the curvature matrix **CUR** may have different magnitudes. To improve the numerical stability, the elements can be scaled by the diagonal elements of the curvature matrix and the damping factor u can be added to the diagonal elements to give the modified curvature matrix **MCUR**



where δkj is the Kronecker delta function (δkj = 1 if *k* = *j* otherwise δkj = 0).

Therefore, we can approximate the incremental changes in the coefficients as

**B** = **MCUR** \* **da**

**da = (MCUR)-1 \* B = MCOV \* B**

where **MCOV** = **MCUR-1** is the **modified covariance matrix.**

The new estimates of the coefficients and the corresponding χ2 value can then be calculated

**anew** = **a** + **da**

To test the minimization χ2 of as part of the iterative process, the following is done

If χ2new > χ2old ⇒ Moving away from a minimum, keep current **a** values and set u = 10 u

Repeat iteration.

If χ2new < χ2old ⇒ Approaching a minimum, set **a** = **anew**, *u* = *u* / 10

Repeat iteration.

⇒ if | χ2new < χ2old | < 0.001

Terminate the iteration

u = 0

χ2 = χ2new

Calculate: **CUR**, **MCUR**, **MCOV**

Step 4: Output the results

The square root of the diagonal elements of the covariance matrix **COV** give the **uncertainties** in the coefficients for **a**



where



Basically, we have set up a hypothesis that our measurements can be described by some analytical function *f*(**a**; *x*). We need to be able to statistically test the hypothesis. This can be done using the value of χ2. χ2 is a measure of the total agreement between our measurements and the hypothesis. It can be assumed that the minimum value of χ2 is distributed according to the χ2 distribution with (*n*-*m*) degrees of freedom. Often the **reduced χ2** value{χ2reduced = χ2/(*n*-*m*)} is quoted as a measure of the **goodness-of-fit**

χ2reduced ~ 1 ⇒ hypothesis is acceptable

χ2reduced << 1 ⇒ the fit is much better than expected given the size of the measurement uncertainties. The hypothesis is acceptable, but the uncertainties σ*y* may have been overestimated.

χ2reduced >> 1 ⇒ hypothesis may not be acceptable

Also the probability of the χ2 value being exceeded is given as another measure of the goodness-of-fit.

Finally, the measurements and fitted function are plotted together with a plot of the χ2 distribution.

**References**

**G. Cowan, *Statistical Data Analysis*, (Clarendon Press, Oxford, 1998)**

Advanced treatment of practical applications of statistics in data analysis.

**A. Kharab and R.B. Guenther, *An Introduction to Numerical Methods – A Matlab Approach,* (Chapman & Hall/CRC, Boca, 2002)**

An introduction to theory and applications in numerical methods for researchers and professionals with a background only in calculus and basic programming. CD-ROM contains the Matlab code for many algorithms

**L. Lyons, *A Practical Guide to Data Analysis for Physical Science Students*, (Cambridge University Press, Cambridge, 1991)**

Basic introduction to analysis of measurement and least squares fitting.

**L. Kirkup, *Experimental Methods*, (John Wiley & Sons, Brisbane, 1994)**

Introduction to the analysis and presentation of experimental data.

**W. R. Leo, *Techniques for Nuclear and Particle Physics Experiments*, (Spring-Verlag, Berlin, 1987)**

Chapter on Statistics and Treatment of Experimental Data including Curve Fitting: linear and non linear fits.

**W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes in C*, (Cambridge University Press, Cambridge, 1992)**

Chapter on modeling of data including linear and non linear fits.

**S. S. M. Wong, *Computational Methods in Physics and Engineering*, (Prentice Hall,** Englewood Cliffs, 1992)

Contains a very good chapter on the Method of Least squares and the Method of Marquardt.