Computational Physics

Ch 04 - Least square fits

Korea University
Eunil Won

The Least-Square Method-I

- The least-square principle

At the observational points $x_1, x_2, \ldots, x_N$ we are given a set of N independent, experimental values $y_1, y_2, \ldots, y_N$. The true values $\eta_1, \eta_2, \ldots, \eta_N$ of the observables are not known, but we assume that some theoretic model exists, which predicts the true value associated with each x_i through some functional dependence,

$$f_i = f_i(\theta_1, \theta_2, ..., \theta_L; x_i) \qquad L \leq N$$

where $\theta_1, \theta_2, \ldots, \theta_L$ is a set of parameters. According to the least-square principle the best values of the unknown parameters are those which make

$$\chi^2 \equiv \sum_{i=1}^{N} w_i (y_i - f_i)^2 = \mathrm{minimum}$$

where ω_i is the weight ascribed to the *i*-th observation. The set of parameters $\underline{\hat{\theta}} = \{\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_L\}$ which produce the smallest value for χ^2 is called *the least square estimates* of the parameters. The procedure of finding the minimum of the chi-square is also called the *least-square fit*.

The Least-Square Method-2

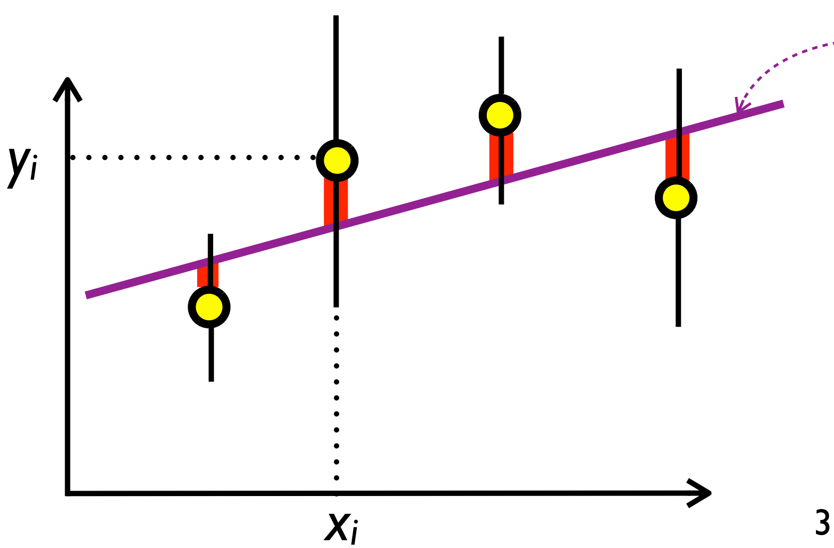
The weight ω_i expresses the accuracy in the measurement y_i. Usually the weight of the *i*-th observation is taken equal to its precision as

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - f_i}{\sigma_i} \right)^2$$

Example) Given $x_1, x_2, \ldots, x_N$ there are $y_1, y_2, \ldots, y_N$ experimental values. Our theoretical model is a straight line:

$$f_i = f_i(\theta_1, \theta_2; x_i) = \theta_1 + \theta_2 x_i$$

If there are errors σ_i for each y_i, the graphical meaning of the chi-square is



$f_i = f_i(\theta_1, \theta_2; x_i) = \theta_1 + \theta_2 x_i$

χ^2 is the square sum of the "distance" between the data and the theory

The Least-Square Method-3

- The least-square fit with histograms

Suppose that the measurement y_i gives the number of events in a class *i*. One could then use the approximation, $\sigma_i^2 \approx f_i$ equivalent to considering a Poisson variable.

this is precisely the case for the histogram we discussed previously. Then the chi-square becomes

$$\chi^2 = \sum_{i=1}^{N} \frac{(y_i - f_i)^2}{f_i}$$

- The least-square fit when observations are correlated.

If observations are correlated, with errors and covariance terms given by the covariance matrix, the least-square principle for finding the best values of the unknown parameters is formulated as

$$\chi^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} (y_i - f_i) V_{ij}^{-1} (y_j - f_j) = \text{minimum}$$

The Least-Square Model-I

- Fitting a straight line - unweighted estimation

$$f_i = \theta_1 + x_i\theta_2$$

If all errors can be neglected, the least-square solution is obtained by determining the minimum of the unweighted sum of squared deviations:

$$\chi^2 = \sum_{i=1}^{N}(y_i - f_i)^2 = \sum_{i=1}^{N}(y_i - \theta_1 - x_i\theta_2)^2$$

$$\frac{\partial\chi^2}{\partial\theta_1} = \sum_{i=1}^{N}(-2)(y_i - \theta_1 - x_i\theta_2) = 0 \qquad \frac{\partial\chi^2}{\partial\theta_2} = \sum_{i=1}^{N}(-2x_i)(y_i - \theta_1 - x_i\theta_2) = 0$$

This set of linear equations can be written in the form

$$N\theta_1 + \sum_{i=1}^{N} x_i\theta_2 = \sum_{i=1}^{N} y_i$$

$$\sum_{i=1}^{N} x_i\theta_1 + \sum_{i=1}^{N} x_i^2\theta_2 = \sum_{i=1}^{N} x_i y_i$$

$$\hat{\theta}_1 = \frac{\sum x_i^2 \sum y_i - \sum x_i y_i \sum x_i}{N\sum x_i^2 - (\sum x_i)^2}$$

$$\hat{\theta}_2 = \frac{N\sum x_i y_i - \sum x_i \sum y_i}{N\sum x_i^2 - (\sum x_i)^2}$$

The Least-Square Model-2

- Fitting a straight line - the normal equation.

How are the independent observations ($y_1 \pm \sigma_1$), ($y_2 \pm \sigma_2$), ..., ($y_N \pm \sigma_N$) at the points $x_1, x_2, ..., x_N$ can be fitted by the weighted least-square method to a linear model with L parameters

$$f_i = f_i(\theta_1, \theta_2, ..., \theta_L; x_i) = \sum_{\ell=1}^{L} a_{i\ell}\theta_\ell, \qquad i = 1, 2, ..., N$$

where $L \leq N$. Here ail is the coefficient appearing with the l-th parameter; usually a_{il} is a function of the x_i's. We now seek the parameter values which minimize

$$\chi^2 = \sum_{i=1}^{N}\left(\frac{y_i - f_i}{\sigma_i}\right)^2 = \sum_{i=1}^{N}\frac{1}{\sigma_i^2}\left(y_i - \left(\sum_{\ell=1}^{L} a_{i\ell}\theta_\ell\right)\right)^2$$

By equating all derivatives to zero we get the L conditions:

$$\frac{\partial\chi^2}{\partial\theta_k} = \sum_{i=1}^{N}(-2a_{ik})\frac{1}{\sigma_i^2}\left(y_i - \sum_{\ell=1}^{L} a_{i\ell}\theta_\ell\right) = 0, \qquad k = 1, 2, ..., L$$

or

$$\sum_{\ell=1}^{L}\left(\sum_{i=1}^{N}\frac{a_{ik}a_{i\ell}}{\sigma_i^2}\right)\theta_\ell = \sum_{i=1}^{N}\frac{a_{ik}y_i}{\sigma_i^2}, \qquad k = 1, 2, ..., L$$

The Least-Square Model-3

If we expand $$\sum_{\ell=1}^{L}\left(\sum_{i=1}^{N}\frac{a_{ik}a_{i\ell}}{\sigma_i^2}\right)\theta_\ell=\sum_{i=1}^{N}\frac{a_{ik}y_i}{\sigma_i^2}, \qquad k=1,2,...,L$$ we get

$$\sum_{i=1}^{N}\frac{a_{i1}a_{i1}}{\sigma_i^2}\theta_1+\sum_{i=1}^{N}\frac{a_{i1}a_{i2}}{\sigma_i^2}\theta_2+...+\sum_{i=1}^{N}\frac{a_{i1}a_{iL}}{\sigma_i^2}\theta_L=\sum_{i=1}^{N}\frac{a_{i1}y_i}{\sigma_i^2}$$

$$\sum_{i=1}^{N}\frac{a_{i2}a_{i1}}{\sigma_i^2}\theta_1+\sum_{i=1}^{N}\frac{a_{i2}a_{i2}}{\sigma_i^2}\theta_2+...+\sum_{i=1}^{N}\frac{a_{i2}a_{iL}}{\sigma_i^2}\theta_L=\sum_{i=1}^{N}\frac{a_{i2}y_i}{\sigma_i^2}$$

$$\vdots$$

$$\sum_{i=1}^{N}\frac{a_{iL}a_{i1}}{\sigma_i^2}\theta_1+\sum_{i=1}^{N}\frac{a_{iL}a_{i2}}{\sigma_i^2}\theta_2+...+\sum_{i=1}^{N}\frac{a_{iL}a_{iL}}{\sigma_i^2}\theta_L=\sum_{i=1}^{N}\frac{a_{iL}y_i}{\sigma_i^2}$$

These are the normal equations for the *L* unknown parameters. Since there are *L* inhomogeneous linear equations for the *L* unknowns the normal equations provide an exact and unique solutions.

The Least-Square Model-4

- Matrix notation.

We rephrase the formulae for the linear problem of the previous notation in terms of matrix notation. We have measurements, theory, and parameters in column vector form:

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}, \quad \mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}, \quad \boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_L \end{pmatrix},$$

The errors in **y** are given in a diagonal *N* by *N* matrix (assuming no correlation)

$$V(\mathbf{y}) = \begin{pmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & & \\ 0 & 0 & \dots & \sigma_N^2 \end{pmatrix}$$

The coefficients are organized in a matrix *A* with *N* rows and L columns

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1L} \\ a_{21} & a_{22} & \dots & a_{2L} \\ \vdots & \vdots & & \\ a_{N1} & a_{N2} & \dots & a_{NL} \end{pmatrix}$$

The Least-Square Model-5

The linear dependence of the theoretical predictions on the parameters

$$f_i = f_i(\theta_1, \theta_2, ..., \theta_L; x_i) = \sum_{\ell=1}^{L} a_{i\ell}\theta_\ell, \qquad i = 1, 2, ..., N$$

is expressed as

$$\mathbf{f} = A\boldsymbol{\theta}$$

and the quantity to be minimized is $\chi^2 = (\mathbf{y} - A\boldsymbol{\theta})^T V^{-1} (\mathbf{y} - A\boldsymbol{\theta})$

Requiring the derivatives equal to 0: $\nabla_{\boldsymbol{\theta}}\chi^2 = -2(A^T V^{-1}\mathbf{y} - A^T V^{-1} A\boldsymbol{\theta}) = 0$

from which $(A^T V^{-1} A)\boldsymbol{\theta} = A^T V^{-1}\mathbf{y}$

and therefore $\boldsymbol{\theta} = (A^T V^{-1} A)^{-1} A^T V^{-1}\mathbf{y}$ is satisfied.

Note: one observes that to find the solution $\boldsymbol{\theta}$, it is sufficient that the covariance matrix $V = V(\mathbf{y})$ is known up to a multiplicative factor.

Now, what about the uncertainties in the linear least square estimate $\boldsymbol{\theta}$ of the parameters?

The Least-Square Model-6

Uncertainties in the linear least square estimate θ of the parameters: we need to calculate the covariance matrix for the least square estimate θ as

$$V(\boldsymbol{\theta}) = \left((A^T V^{-1} A)^{-1} A^T V^{-1}\right) V(\mathbf{y}) \left((A^T V^{-1} A)^{-1} A^T V^{-1}\right)^T$$

Where is this from?

In the error propagation of ch01, we have the transformation property of the covariance matrix as $U = AVA^T$

The above expression can be simplified to $V(\boldsymbol{\theta}) = (A^T V^{-1} A)^{-1}$

(Can you prove it?)

The Least-Square Model-example

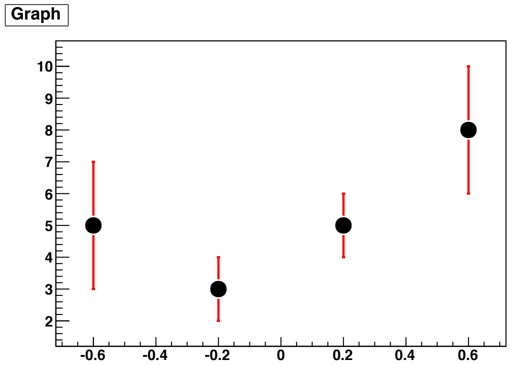
- Fitting a parabola

We want to fit the parabolic parameterization

$$f(\theta_1, \theta_2, \theta_3; x) = \theta_1 + \theta_2 x + \theta_3 x^2$$

to the following observations:

i	1	2	3	4
x_i	-0.6	-0.2	0.2	0.6
$y_i \pm \sigma_i$	5 ± 2	3 ± 1	5 ± 1	8 ± 2



Determine parameters **θ** of the parabola. (3 unknowns and four observations)

The matrix *A* (4x3) and the independent measurements define the column vector **y** = (5,3,5,8) and the diagonal covariance matrix

$$A = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \end{pmatrix} = \begin{pmatrix} 1 & -0.6 & (-0.6)^2 \\ 1 & -0.2 & (-0.2)^2 \\ 1 & 0.2 & 0.2^2 \\ 1 & 0.6 & 0.6^2 \end{pmatrix} \qquad V = \begin{pmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{pmatrix} = \begin{pmatrix} 2^2 & 0 & 0 & 0 \\ 0 & 1^2 & 0 & 0 \\ 0 & 0 & 1^2 & 0 \\ 0 & 0 & 0 & 2^2 \end{pmatrix}$$

The resulting matrix product that we want is:

$$\begin{aligned} A^T V^{-1} A &= \begin{pmatrix} 1 & 1 & 1 & 1 \\ -0.6 & -0.2 & 0.2 & 0.6 \\ (-0.6)^2 & (-0.2)^2 & 0.2^2 & 0.6^2 \end{pmatrix} \begin{pmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.25 \end{pmatrix} \begin{pmatrix} 1 & -0.6 & (-0.6)^2 \\ 1 & -0.2 & (-0.2)^2 \\ 1 & 0.2 & 0.2^2 \\ 1 & 0.6 & 0.6^2 \end{pmatrix} \\ &= \begin{pmatrix} 2.5 & 0 & 0.26 \\ 0 & 0.26 & 0 \\ 0.26 & 0 & 0.068 \end{pmatrix} \end{aligned}$$

The Least-Square Model-example

We need to invert the previous matrix: $$(A^TV^{-1}A)^{-1} = \begin{pmatrix} 0.664 & 0 & -2.54 \\ 0 & 3.85 & 0 \\ -2.54 & 0 & 24.42 \end{pmatrix}$$

We find the least square solution **θ** by carrying out the multiplication of the matrices

$$\begin{aligned} \boldsymbol{\theta} &= (A^TV^{-1}A)^{-1}A^TV^{-1}\mathbf{y} \\ &= \begin{pmatrix} 0.664 & 0 & -2.54 \\ 0 & 3.85 & 0 \\ -2.54 & 0 & 24.42 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ -0.6 & -0.2 & 0.2 & 0.6 \\ (-0.6)^2 & (-0.2)^2 & 0.2^2 & 0.6^2 \end{pmatrix} \begin{pmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.25 \end{pmatrix} \begin{pmatrix} 5 \\ 3 \\ 5 \\ 8 \end{pmatrix} \\ &= \begin{pmatrix} 3.68 \\ 3.27 \\ 7.81 \end{pmatrix} \end{aligned}$$

$$\therefore \mathbf{f} = \mathbf{f}(\boldsymbol{\theta}; x) = 3.68 + 3.27x + 7.81x^2$$

The accuracy of the estimated parameters can be found from the covariance matrix V(**θ**). Hence the estimates of the errors are given by the square roots of the diagonal elements of this matrix

$$(A^TV^{-1}A)^{-1} = \begin{pmatrix} 0.664 & 0 & -2.54 \\ 0 & 3.85 & 0 \\ -2.54 & 0 & 24.42 \end{pmatrix}$$

$$\sigma_{\theta_1} = \sqrt{0.664} = 0.81, \quad \sigma_{\theta_2} = \sqrt{3.85} = 1.96, \quad \sigma_{\theta_3} = \sqrt{24.42} = 4.94,$$

The Least-Square Model-example

Parameters are correlated. For example,

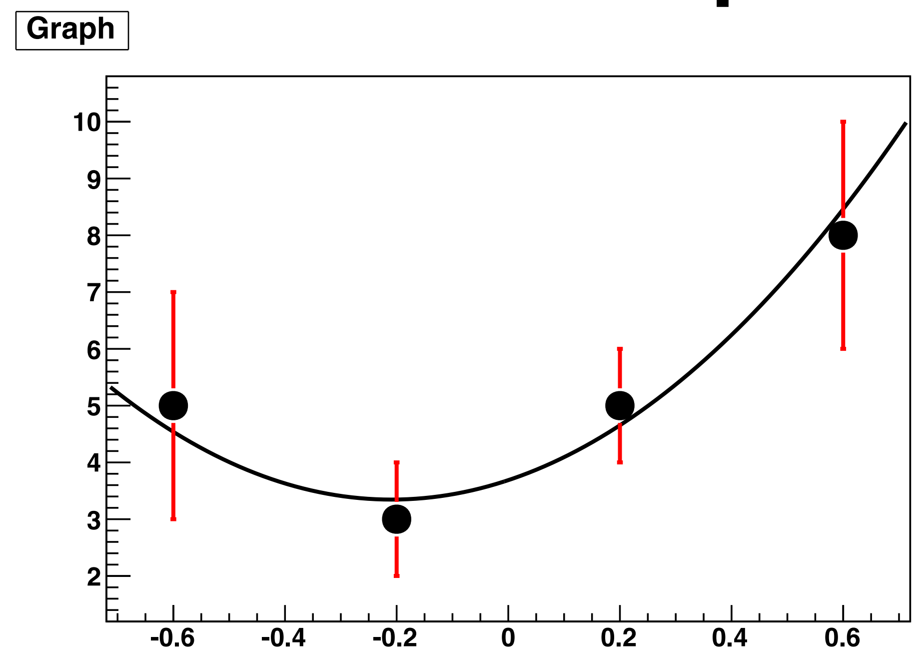
$$\rho_{13} = \frac{(A^T V^{-1} A)^{-1}_{13}}{\sigma_{\theta_1} \sigma_{\theta_3}} = \frac{-2.54}{0.81 \cdot 4.94} = -0.63$$

Can you understand why they are correlated?

The Least-Square Model-example

How can I do this fit in ROOT?

```
//
// a parabola fit - minimal invocation of MINUIT
//
// Eunil Won (eunil@hep.korea.ac.kr)
//
Double_t f1(Double_t *x, Double_t *par)
{
  return par[0] + par[1]*x[0] + par[2]*x[0]*x[0];
}

void para()
{
   gROOT->SetStyle("Plain");
   gROOT->ForceStyle();

   Int_t n = 4;
   Double_t *x1 = new Double_t[n];
   Double_t *y1 = new Double_t[n];
   Double_t *e1 = new Double_t[n];

   // save the data
   x1[0] = -0.6; x1[1] = -0.2; x1[2] =  0.2; x1[3] =  0.6;
   y1[0] =  5.0; y1[1] =  3.0; y1[2] =  5.0; y1[3] =  8.0;
   e1[0] =  2.0; e1[1] =  1.0; e1[2] =  1.0; e1[3] =  2.0;

   TF1 *func = new TF1("func",f1,-1.0,1.0,3);

   //create the graphs and set their drawing options
   TGraphErrors *gr1 = new TGraphErrors(n, x1, y1, 0, e1);
   gr1->SetLineColor(kRed);
   gr1->SetLineWidth(3.0);
   gr1->SetMarkerStyle(20);
   gr1->SetMarkerSize(3.0);

   TCanvas *myc = new TCanvas("myc","Fitting a parabola");
   gr1->Draw("ap");
   gr1->Fit("func");
}
```




```
root -l para.C
root [0]
Processing para.C...
 FCN=0.346154 FROM MIGRAD    STATUS=CONVERGED      50 CALLS          51 TOTAL
                     EDM=3.95683e-24    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           3.68750e+00   8.14900e-01   3.58300e-04   4.26055e-12
   2  p1           3.26923e+00   1.96116e+00   1.11104e-03   3.74723e-13
   3  p2           7.81250e+00   4.94106e+00   2.17252e-03   5.11031e-13
root [1] .q
```

estimates   errors

: are these values same as the results from the analytic calculation?

The Least-Square Model-example

In the previous example, the minimization is implicitly done by calling

```
gr1->Fit("func");
```

with the definition of f(θ;x) is defined as

```
Double_t f1(Double_t *x, Double_t *par)
{
  return par[0] + par[1]*x[0] + par[2]*x[0]*x[0];
}
```

So, the minimization is done somewhere. Who is doing it?
: the answer is - a package called MINUIT is used by ROOT.

OoooK, then what is MINUIT?

MINUIT is a tool to find the minimum values of a multi-parameter function and analyze the shape of the function around the minimum.
(Online manual is available from http://wwwasdoc.web.cern.ch/wwwasdoc/minuit/minmain.html)

```
root -l para.C
root [0]
Processing para.C...
 FCN=0.346154 FROM MIGRAD    STATUS=CONVERGED      50 CALLS          51 TOTAL
                     EDM=3.95683e-24    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           3.68750e+00   8.14900e-01   3.58300e-04   4.26055e-12
   2  p1           3.26923e+00   1.96116e+00   1.11104e-03   3.74723e-13
   3  p2           7.81250e+00   4.94106e+00   2.17252e-03   5.11031e-13
root [1] .q
```

These are the outputs from MINUIT package, in ROOT

You don't need to understand fully what MINUIT is, but at least should understand the routines that appear in the class!

The Least-Square Model-example

More elaborated way:

```
#include "TMinuit.h"

gROOT->SetStyle("Plain");
gROOT->ForceStyle();

Int_t n = 4;
Double_t *x1 = new Double_t[n];
Double_t *y1 = new Double_t[n];
Double_t *e1 = new Double_t[n];

x1[0] = -0.6; x1[1] = -0.2; x1[2] =  0.2; x1[3] =  0.6;
y1[0] =  5.0; y1[1] =  3.0; y1[2] =  5.0; y1[3] =  8.0;
e1[0] =  2.0; e1[1] =  1.0; e1[2] =  1.0; e1[3] =  2.0;

Double_t func(Double_t x, Double_t *par)
{
  return par[0] + par[1]*x + par[2]*x*x;
}

void fcn(Int_t &npar, Double_t *gin, Double_t &f, Double_t *par, Int_t iflag)
{
  Int_t i;

  Double_t chisq = 0;
  Double_t delta;

  for (i=0;i<n; i++) {
   delta  = (y1[i]-func(x1[i],par))/e1[i];
   chisq += delta*delta;
  }
  f = chisq;
}
```

Now data taken out to be global variables

`fcn()` *is the function to be minimized*

```
void para_minuit()
{
   //
   // calls MINUIT explicitly
   //
   TMinuit *gMinuit = new TMinuit(3);  // initialize with a maximum of 3 params
   gMinuit->SetFCN(fcn);               // set the minimization function
   Double_t arglist[10];
   Int_t ierflg = 0;

   arglist[0] = 1;
   gMinuit->mnexcm("SET ERR", arglist ,1,ierflg);

   static Double_t vstart[3] = {0.0, 0.0, 0.0};
   static Double_t step[3] = {0.001, 0.001, 0.001};
   gMinuit->mnparm(0, "theta_0", vstart[0], step[0], 0,0,ierflg);
   gMinuit->mnparm(1, "theta_1", vstart[1], step[1], 0,0,ierflg);
   gMinuit->mnparm(2, "theta_2", vstart[2], step[2], 0,0,ierflg);

   arglist[0] = 5000;
   arglist[1] = 1.;
   gMinuit->mnexcm("MINUIT", arglist ,2,ierflg);
}
```

`para_minuit()` *contains the explicit control of the MINUIT functions*

```
 FCN=0.346154 FROM MIGRAD    STATUS=CONVERGED      56 CALLS          57 TOTAL
                     EDM=3.71822e-23    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  theta_0      3.68750e+00   8.14900e-01   3.58300e-04  -1.27816e-11
   2  theta_1      3.26923e+00   1.96116e+00   1.11104e-03   8.74355e-13
   3  theta_2      7.81250e+00   4.94106e+00   2.17252e-03  -8.30426e-13
 EXTERNAL ERROR MATRIX.    NDIM=  25    NPAR=  3    ERR DEF=1
  6.641e-01  1.130e-11 -2.539e+00
  1.130e-11  3.846e+00  7.111e-09
 -2.539e+00  7.111e-09  2.441e+01
 PARAMETER  CORRELATION COEFFICIENTS
       NO.  GLOBAL      1      2      3
        1  0.63059   1.000  0.000 -0.631
        2  0.00000   0.000  1.000  0.000
        3  0.63059  -0.631  0.000  1.000
```

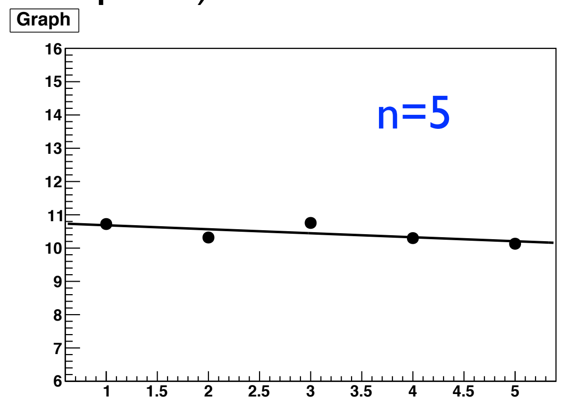
Same fit results

$$(A^TV^{-1}A)^{-1} = \begin{pmatrix} 0.664 & 0 & -2.54 \\ 0 & 3.85 & 0 \\ -2.54 & 0 & 24.42 \end{pmatrix}$$

$$\rho_{13} = \frac{(A^TV^{-1}A)^{-1}_{13}}{\sigma_{\theta_1}\sigma_{\theta_3}} = \frac{-2.54}{0.81 \cdot 4.94} = -0.63$$

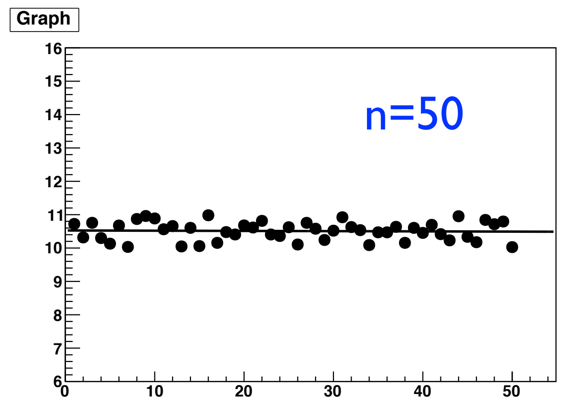
Error vs. number of points? 1

Let's examine the size of error and the number of points in the fit, empirically
: generate n random numbers of 10+ r_i and fits to a linear polynomial (no error is assigned on each data point)



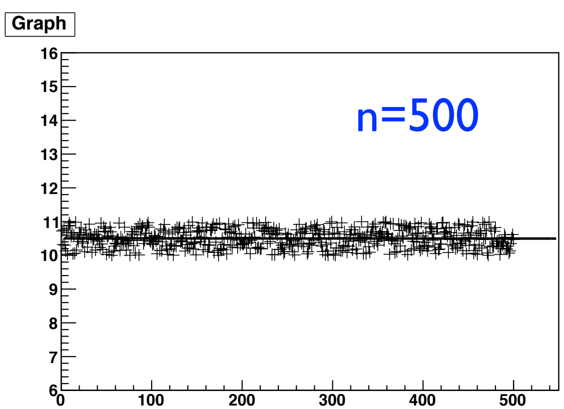

```
FCN=0.165204 FROM MIGRAD    STATUS=CONVERGED      31 CALLS          32 TOTAL
                     EDM=5.6398e-24    STRATEGY= 1      ERROR MATRIX ACCURATE
 EXT PARAMETER                                   STEP         FIRST
 NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
  1  p0           1.08059e+01   1.04881e+00   2.35714e-04  -4.71004e-12
  2  p1          -1.20102e-01   3.16228e-01   7.10706e-05  -5.85803e-12
```

$$\sigma_{\text{offset}} = 1.0$$




```
FCN=3.79088 FROM MIGRAD    STATUS=CONVERGED      33 CALLS          34 TOTAL
                     EDM=1.66744e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
 EXT PARAMETER                                   STEP         FIRST
 NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
  1  p0           1.05268e+01   2.87139e-01   1.51145e-04   1.24872e-10
  2  p1          -6.79418e-04   9.79992e-03   5.15849e-06   3.65878e-09
```

$$\sigma_{\text{offset}} = 0.29$$




```
FCN=38.3934 FROM MIGRAD    STATUS=CONVERGED      35 CALLS          36 TOTAL
                     EDM=1.01478e-19    STRATEGY= 1      ERROR MATRIX ACCURATE
 EXT PARAMETER                                   STEP         FIRST
 NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
  1  p0           1.05017e+01   8.95770e-02   1.37056e-04   9.97986e-09
  2  p1          -9.48712e-06   3.09839e-04   4.74063e-07   2.69790e-06
```

$$\sigma_{\text{offset}} = 0.090$$

So, obviously, more data means smaller error!

Error vs. number of points? 2

Q: Can you explain and expect exact behavior of the error as a function of the number of data?

A: It should be from the relation we derived before - $V(\boldsymbol{\theta}) = (A^T V^{-1} A)^{-1}$

In our example, we have two fit parameters and N data points. So,

$$V(\boldsymbol{\theta}) = \left(\begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_N \end{pmatrix} \begin{pmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 0 & \sigma_N^2 \end{pmatrix}^{-1} \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{pmatrix} \right)^{-1}$$

For simplicity, we assume $\sigma_1 = \sigma_2 = .. = \sigma_N$, and all x_i are equally spaced ($x_3 = x_2 + \delta = x_1 + 2\delta$ etc), then

$$\begin{aligned} V(\boldsymbol{\theta}) &= \left(\begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_1+\delta & \dots & x_1+(N-1)\delta \end{pmatrix} \frac{1}{\sigma^2} \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & x_1 \\ 1 & x_1+\delta \\ \vdots & \vdots \\ 1 & x_1+(N-1)\delta \end{pmatrix} \right)^{-1} \\ &= \sigma^2 \begin{pmatrix} N & Nx_1 + \frac{N(N-1)}{2}\delta \\ Nx_1 + \frac{N(N-1)}{2}\delta & N \end{pmatrix}^{-1} \\ &= \frac{\sigma^2}{N^2 - \left(Nx_1 + \frac{N(N-1)}{2}\delta\right)^2} \begin{pmatrix} N & -Nx_1 - \frac{N(N-1)}{2}\delta \\ -Nx_1 - \frac{N(N-1)}{2}\delta & N \end{pmatrix} \end{aligned}$$

Error vs. number of points? 3

So, the square-root of the diagonal terms which are the error of the fit parameters, are proportional to

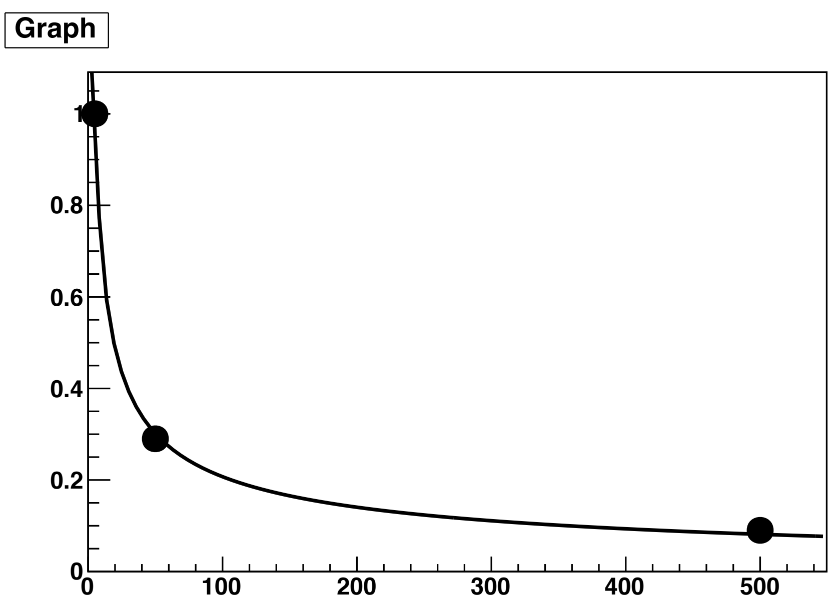
$$\sigma_{\text{offset}} \propto \frac{\sigma}{\sqrt{N}}$$

Coming back to our example:

n=5 $\sigma_{\text{offset}} = 1.0$

n=50 $\sigma_{\text{offset}} = 0.29$

n=500 $\sigma_{\text{offset}} = 0.090$




```
Double_t f1(Double_t *x, Double_t *par)
{
  Double_t val;

  if ( x[0] > 0 )
  {
    val = par[0] + par[1]/TMath::Sqrt(x[0]);
  } else val = par[0];
  return val;
}

void error()
{
   gROOT->SetStyle("Plain");
   gROOT->ForceStyle();

   Int_t n = 3;
   Double_t *x1 = new Double_t[n];
   Double_t *y1 = new Double_t[n];
   Double_t *e1 = new Double_t[n];

   // save the data
   x1[0] =  5.0; x1[1] = 50.0; x1[2] = 500.0;
   y1[0] =  1.0; y1[1] = 0.29; y1[2] = 0.090;

   TF1 *func = new TF1("func",f1,1.0,600,2);

   //create the graphs and set their drawing options
   TGraphErrors *gr1 = new TGraph(n, x1, y1);
   gr1->SetLineWidth(3.0);
   gr1->SetMarkerStyle(20);
   gr1->SetMarkerSize(3.0);

   TCanvas *myc = new TCanvas("myc","Fitting 1/sqrt(x)");
   gr1->Draw("ap");
   gr1->Fit("func");
}
```

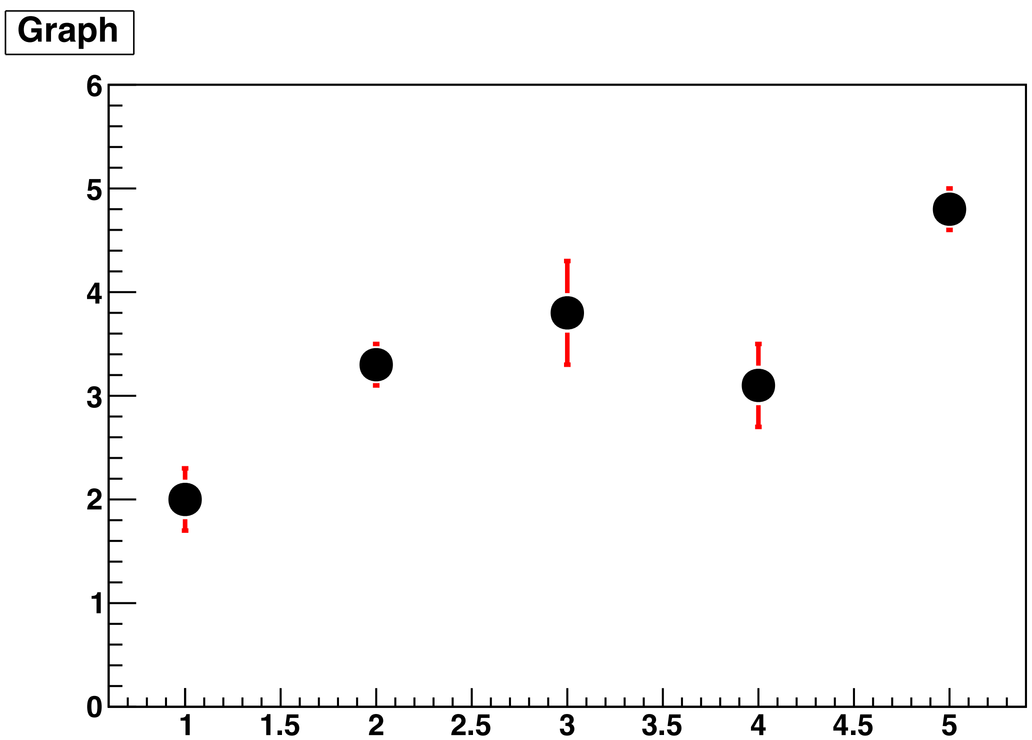
Fit agrees well with our expectation!

Least squares fit of a polynomial-1

Suppose I have a set of 5 data, with errors given as

i	1	2	3	4	5
x_i	1	2	3	4	5
$y_i \pm \sigma_i$	2 ± 0.3	3.3 ± 0.2	3.8 ± 0.5	3.1 ± 0.4	4.8 ± 0.2

So it looks like



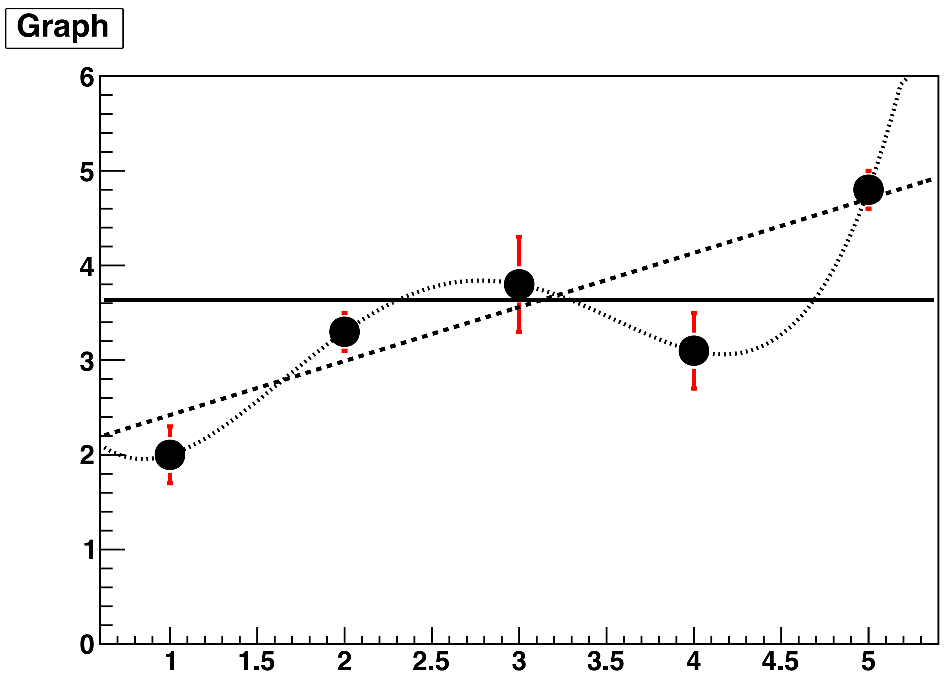
Which degree of polynomial that I want to use?

- 0th
- 1st
- 4th order?

How can we decide?

Let's fit them with all three options anyway!

Least squares fit of a polynomial-2



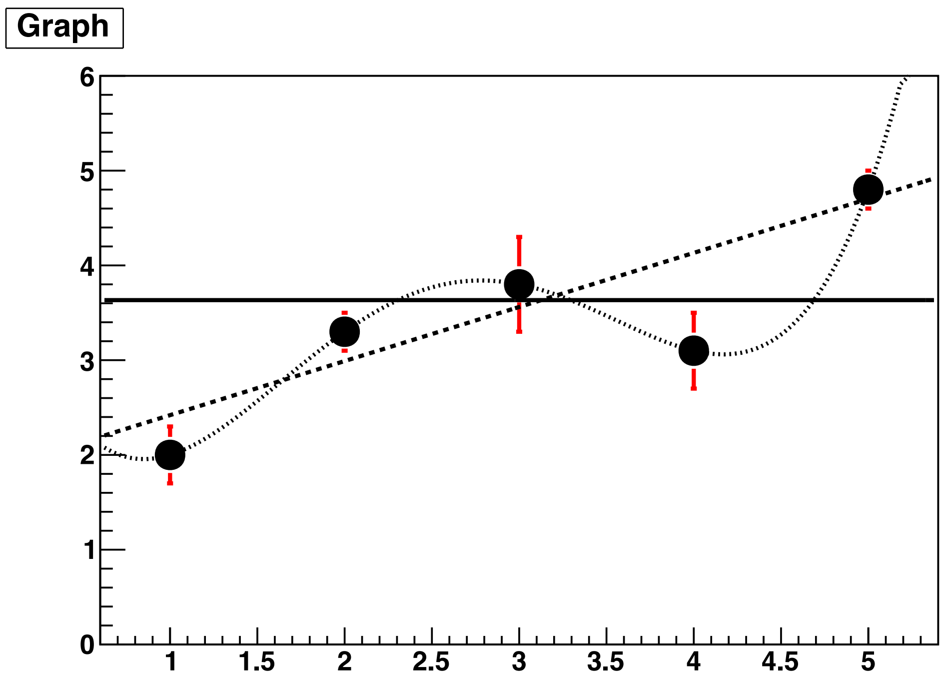

```
Processing degree.C...
FCN=68.3364 FROM MIGRAD    STATUS=CONVERGED      12 CALLS          13 TOTAL
                     EDM=8.92491e-17    STRATEGY= 1      ERROR MATRIX
ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           3.63359e+00   1.18378e-01   4.81304e-04   1.12862e-07
 FCN=11.4733 FROM MIGRAD    STATUS=CONVERGED      31 CALLS          32 TOTAL
                     EDM=4.97127e-22    STRATEGY= 1      ERROR MATRIX
ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.84911e+00   2.64601e-01   2.04141e-04  -8.70163e-11
   2  p1           5.70759e-01   7.56898e-02   5.83948e-05   1.21679e-10
 FCN=4.1015e-14 FROM MIGRAD    STATUS=CONVERGED     153 CALLS         154
TOTAL
                     EDM=6.29991e-14    STRATEGY= 1      ERR MATRIX NOT POS-
DEF
  EXT PARAMETER                APPROXIMATE        STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           4.30000e+00   4.56553e-01   5.78015e-05  -4.38159e-07
   2  p1          -6.56667e+00   3.28004e-01   1.65342e-05   3.22706e-06
   3  p2           5.83333e+00   8.92122e-02   3.64347e-06   2.43198e-05
   4  p3          -1.73333e+00   1.85889e-02   8.26518e-07   1.37390e-04
   5  p4           1.66667e-01   2.97512e-03   1.52828e-07   7.26206e-04
```

Chi-square values are available as “FCN” in the MINUIT output (see the text above carefully)

0-th order: 68.3
1st order: 11.5
4th order: 4.1e-14 (practically 0.0)

Since the chi-square is the distance between data and model, it should in principle be better for smaller values. Is 4th order polynomial fit is the best then?

Least squares fit of a polynomial-3



For the 4th order fit, the fit function passes all the data points: the fit function has "too much" flexibility...

We need to think of the "degree of freedom" of the chi-square in each fits:

0-th order: 5 data points and 1 parameter in the fit function
1st order: 5 data points and 2 parameters
4th order: 5 data points and 5 parameters

The number of degrees of freedom (n.d.f) : the number of data points - the number of parameters

0-th order: 5-1 = 4
1st order: 5-2 = 3
4th order: 5-5 = 0

When the degrees of freedom is 0, the chi-square values is meaningless at all - so we don't discuss when the degrees of freedom is zero.

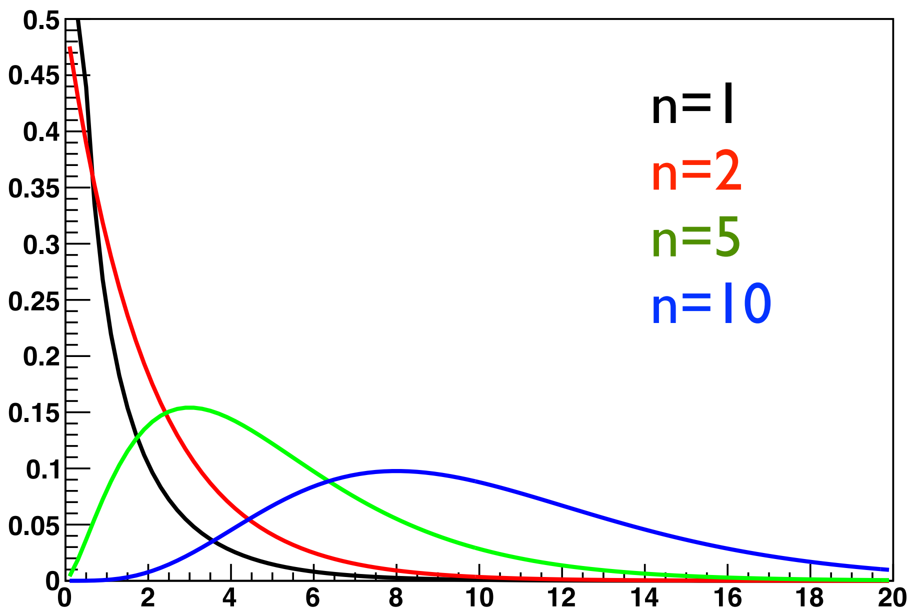
Goodness-of-fit with chi-square 1

Except when n.d.f. = 0, chi-square must tell us how good the given fit is. But we need to take into account n.d.f. somehow.

The distribution of chi-square $f(\chi^2)$? For random variables, the distribution is known to be

$$f(\chi^2) = f(z; n)$$

where $$f(z;n) = \frac{1}{2^{n/2}\Gamma(n/2)} z^{n/2-1} e^{-z/2}, \quad n = 1, 2, \ldots \quad (0 \le z < \infty)$$



We discussed this in Ch02 (Examples of distributions)

This means, if n=1, most of the time, the chi-square/n.d.f. should be more or less ~1.0 for the most of the time.

Note that:

$$\begin{aligned} E[z] &= \int_0^\infty z \frac{1}{2^{n/2}\Gamma(n/2)} z^{n/2-1} e^{-z/2}\, dz = n, \\ V[z] &= \int_0^\infty (z-n)^2 \frac{1}{2^{n/2}\Gamma(n/2)} z^{n/2-1} e^{-z/2}\, dz = 2n \end{aligned}$$

Goodness-of-fit with chi-square 2

$f(\chi^2)/\mathrm{n.d.f.}$ is a measure of goodness-of-fit.

- If $f(\chi^2)/\mathrm{n.d.f.}$ is near one, then all is as expected.

- If $f(\chi^2)/\mathrm{n.d.f.}$ is much less than one, then the fit is better than expected given the size of the measurement errors: this is not bad in the sense of providing evidence against the hypothesis, but it is usually grounds to check that the errors σ_i have not been overestimated or are not correlated.

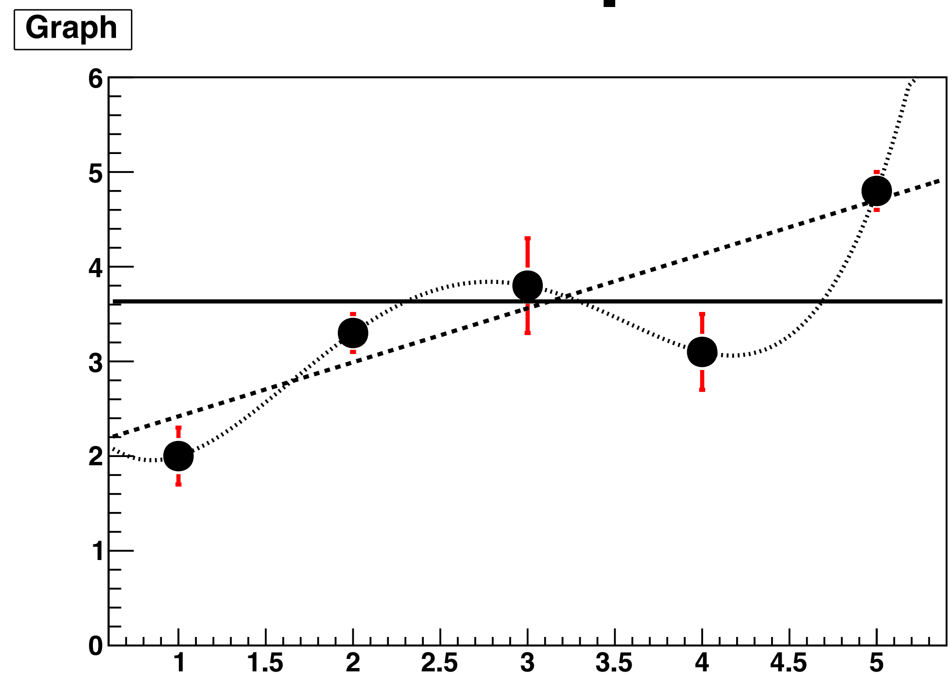
- If $f(\chi^2)/\mathrm{n.d.f.}$ is much larger than one, then there is some reason to doubt the hypothesis.

One often quotes a significance level (p-value) for a given χ^2, which is the probability that the hypothesis would lead to a χ^2 value worse (i.e. greater) than the one actually obtained.

$$p = \int_{\chi^2}^{\infty} f(z;n)dz$$

Goodness-of-fit with chi-square 3

Let us come back to our example.



Chi-square values and n.d.f. are

	χ^2	n.d.f.
0th-order	68.3	4
1st-order	11.5	3

1) For the 1st-order case, the p-value is $9.3\text{x}10^{-3}$

That is, if the true function were a straight line and if the experiment were repeated many times, one would expect the χ^2 values to be worse (i.e. higher) than the one actually obtained (χ^2 =11.5) in $9.3\text{x}10^{-3}$ of the case: so the straight line hypothesis may not be true?

2) For the 0th-order case, the p-value is $5.2\text{x}10^{-14}$

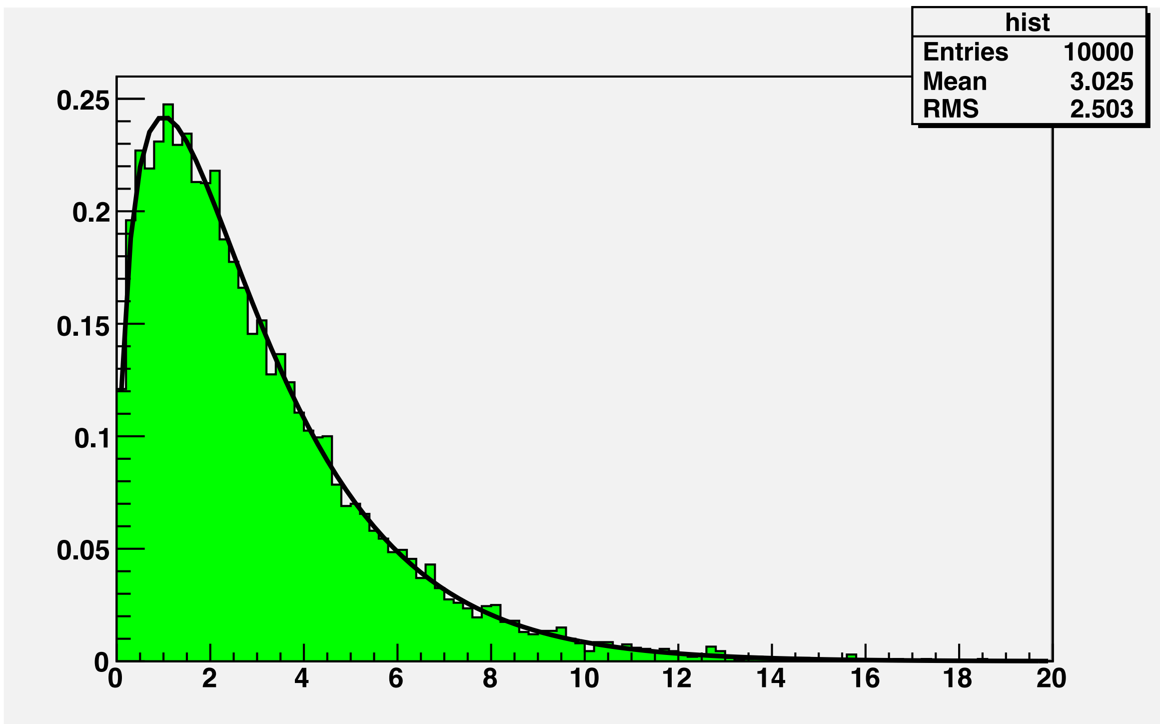
If the horizontal-line hypothesis were true, one would expect a χ^2 value as high or higher than the one one obtained in only 5 times out of 10^{14} trials: so this hypothesis can safely be ruled out!

By the way, how can I compute above p-values in ROOT?

```
root [0] TMath::Prob(11.5,3)
(Double_t)9.30779710609636393e-03
```

Goodness-of-fit with chi-square 4

Demonstration of the chi-square distribution:



line: function drawing of chi-square when n=3

histogram: distribution of chi-squares for 10000 independent fits

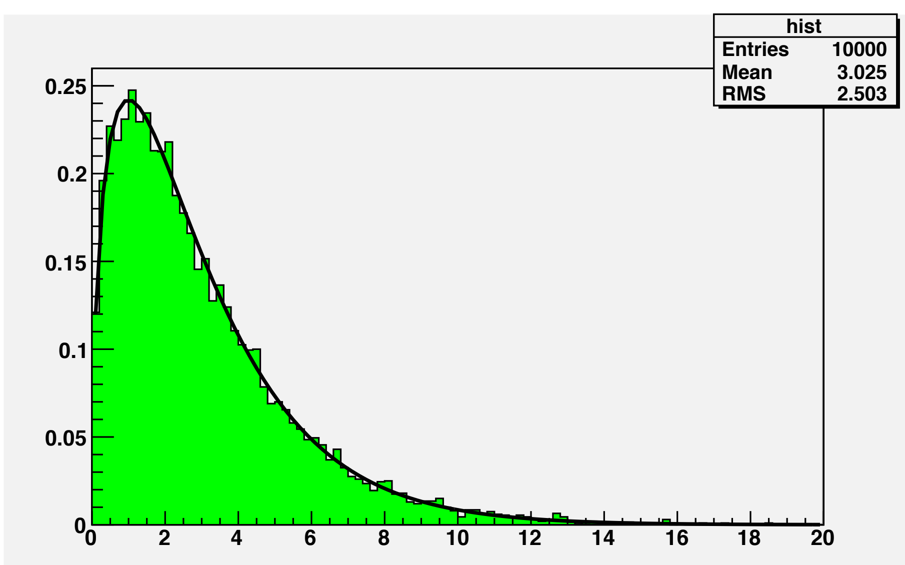
They agree well each other!

Can you write a ROOT program that produces the above plot?
I'll give you only the algorithm this time (see the next slide)

Goodness-of-fit with chi-square 5

Demonstration of the chi-square distribution: my algorithm
(Of course this is not the only way of doing this)

1) Generate *N* random numbers r_i in [0,1]
2) Divide [0,1] into *n* bins
3) Count the number r_i belongs to each bin and save to y_i
4) Error on each bin is e_i = sqrt(y_i)
5) Perform a 0th-order polynomial fit on $\{y_i, e_i\}$
6) Get the chi-square of the fit in 5) and save it into your histogram
7) Repeat 1) - 6) M times and draw the histogram
8) Draw the chi-square function of n.d.f=*n-1* over the histogram



N = 1000
n = 4
M = 10000 in my case of the figure left

Q: How can I get the value of χ^2 after the fit? (amin in the below codes is the one)

```
Double_t amin,edm,errdef;
Int_t nvpar,nparx,icstat;
gMinuit->mnstat(amin,edm,errdef,nvpar,nparx,icstat);
```