Statistical Methods for Particle Physics
Graduierten-Kolleg RWTH Aachen, February 2014
Problem sheet 3

Exercise 1: Consider N independent observations $n_1, \dots, n_N$ of a Poisson random variable with the same unknown mean value ν.

(a) Write down the likelihood function for the parameter ν. (Since the Poisson distribution is not a pdf but rather a probability, here the likelihood is found directly from the joint probability for the data.) Find the maximum-likelihood estimator for ν.

(b) Show that the estimator is unbiased and find its variance in closed form (use the known mean and variance of a Poisson variable).

(c) Show that the variance of $\hat{\nu}$ is equal to the minimum variance bound (the right-hand side of the information inequality).

Exercise 2: This exercise provides an introduction to the class `TMinuit`, used in `ROOT` for function minimization. You should turn in any code where you have made modifications and program output in the form of numerical values and plots as appropriate. You do not need to turn in code supplied to you that you did not modify.

The exercises uses `TMinuit` to carry out a Maximum Likelihood fit where we minimize the quantity $-2 \ln L$. For more information on TMminuit see

`root.cern.ch/root/html/TMinuit.html`

First we will generate some data using a simple Monte Carlo program. Download, build and test the program `makeData` from the course website. `makeData` generates values according to an exponential distribution

$$f(x;\xi) = \frac{1}{\xi} e^{-x/\xi} \quad (x \geq 0)$$

and writes the values to a file.

In a separate directory, download and build the program `expFit` from the course website. This program reads in the file of individual values provided by `makeData` and does a maximum likelihood fit of the parameter ξ nof the exponential pdf. Run `makeData` and generate a file with 200 data values. Use this as the input for `expFit` and find the estimate $\hat{\xi}$ and its standard deviation $\sigma_{\hat{\xi}}$.

Now modify `makeData` so that it generates values according to the pdf

$$f(x;\alpha,\xi_1,\xi_2) = \alpha \frac{1}{\xi_1} e^{-x/\xi_1} + (1-\alpha) \frac{1}{\xi_2} e^{-x/\xi_2} , \qquad (1)$$

with $\alpha = 0.2$, $\xi_1 = 1.0$ and $\xi_2 = 5$. To do this, first generate a random number r uniform in $[0, 1]$. If $r < \alpha$, then generate x according to an exponential with mean ξ_1, otherwise use ξ_2. Run the program and save 200 individual values to a text file.

2 Now modify the program `expFit` so that it reads in the values and carries out an ML fit of the parameters α, ξ_1 and ξ_2. You will have to supply start values and "step sizes" for the parameters. Choose start values not too far (say, within a factor of two) to the true values used in `makeData`. For the step sizes you can take, e.g., 0.1.

Try running the program with the minimum and maximum values (in the arrays `minVal` and `maxVal`) set equal to zero; this is equivalent to having no bounds on the parameters. If the program runs into a region of parameter space that it shouldn't, e.g., $\xi_1 < 0$, then you can place appropriate bounds on the parameter values. In the end it is best to see if you can rerun the fit with improved guesses for start values but without any bounds on the parameters.

Modify the program so it makes a reasonable plot of the fit (extend the limit of the horizontal axis as appropriate). Find the ML estimators and their covariance matrix using the routines `mnpout` and `mnemat`. This requires that you add lines of the form

```
double covmat[npar][npar];
minuit.mnemat(&covmat[0][0], npar);
```

where `npar` (here 3) is the number of fitted parameters. Determine as well the matrix of correlation coefficients.

Exercise 3: For this problem refer to the root macro `SimpleFit.C` and the data file `testData.txt` from the course website. The code is basically C++, but it is executed through the program root rather than being run as an independent program. First run root and at the prompt, type

```
.L simpleFit.C
simpleFit()
```

The first command loads the contents of the file `simpleFit.C` and thus defines the functions contained in it. The second command calls the function `simpleFit()`. This prompts the user for a data file containing columns of numbers representing here the usual ingredients of a least-squares fit, x, y and σ.

The macro contains a fit function, which is currently set up as a polynomial,

$$f(x;\boldsymbol{\theta}) = \sum_{k=0}^{n} \theta_k x^k \ .$$

After reading in the data, the parameters of the polynomial are fitted using the method of least squares. The results are extracted and displayed, including the fitted parameter values, their standard deviations, the minimized χ^2, the corresponding p-value, the covariance matrix $V_{ij} = \mathrm{cov}[\hat{\theta}_i, \hat{\theta}_j]$ and its inverse.

3(a) Try different orders for the polynomial by changing the variable `npar` (i.e., to obtain an order $n =$ `npar - 1`). What is the smallest order giving a p-value greater than 0.1?

3(b) Consider the cases of order $n = 2$, 3 and 4 (i.e., 3, 4 and 5 parameters). By using the polynomial together with the fitted parameters, find the predicted value of the function at $x = 5$, $x = 6$ and $x = 10$. The program contains an object called `f` of type `TF1*` that you can use to access the fit function. You will need (in a loop over the parameters)

```
f->SetParameter(i, thetaHat[i]);
```

to set the parameter values to the fitted ones and

```
f->Eval([x[i]);
```

to evaluate the function at the point `x[i]`. Please refer to the root documentation for more details.

Using error propagation, find the standard deviations of the differences

$$\begin{aligned} d_1 &= f(x=6;\hat{\boldsymbol{\theta}}) - f(x=5;\hat{\boldsymbol{\theta}}) \ , \\ d_2 &= f(x=10;\hat{\boldsymbol{\theta}}) - f(x=5;\hat{\boldsymbol{\theta}}) \ . \end{aligned}$$

Comment on how one expects the variance in the difference to behave as the difference between the x values decreases.

3(c) Suppose for the case of order $n = 3$, a certain model predicts the parameter values: $\theta_0 = -0.75$, $\theta_1 = 2.5$, $\theta_2 = -0.5$ and $\theta_3 = 0.026$. By using the inverse of the covariance matrix found in the macro (variable `Vinv`), compute the χ^2 statistic comparing your fitted values with the model predictions,

$$\chi^2 = \sum_{i,j=0}^{n} (\theta_{\mathrm{mod},i} - \hat{\theta}_i)(V^{-1})_{ij}(\theta_{\mathrm{mod},j} - \hat{\theta}_j)$$

Note that here the estimators $\hat{\boldsymbol{\theta}}$ are treated as a set of 4 measured quantities and they are compared to 4 fixed model predictions. Find the corresponding p-value. Is the model in acceptable agreement with the estimated parameter values?

For purposes of comparison, try computing the χ^2 using only the diagonal elements of the covariance matrix (note this is incorrect!). That is, use the formula

$$\chi^2_{\mathrm{bad}} = \sum_{i=0}^{n} \frac{(\theta_{\mathrm{mod},i} - \hat{\theta}_i)^2}{V[\hat{\theta}_i]} \ .$$

Find the corresponding p-value under the (false) assumption that the statistic follows a chi-square pdf and notice that it leads to a completely erroneous conclusion.

Finally compute the χ^2 by comparing the measured (x, y, σ) values to the prediction $f(x;\boldsymbol{\theta}_{\mathrm{mod}})$. This will not give exactly the same value as obtained from the fitted parameters but it should lead to the same conclusion.

G. Cowan
February 2014