Computing and Statistical Data Analysis
Problem sheet #8

Exercise 1: Consider N independent observations $n_1, \ldots, n_N$ of a Poisson random variable with the mean values $E[n_i] = c_i\nu$ where c_i are known constants and ν is the unknown parameter that we want to estimate.

(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. In this example we will use `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 and writes the values to a file. It also produces a histogram of the values.

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 of the exponential pdf. Run both programs and make sure you understand what they are doing.

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

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

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.

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`. Determine as well the matrix of correlation coefficients.

Optional: As an extension to this problem, convert the fitting program to read in the histogram of generated values, rather than each individual value. Then construct the likelihood function using

$$\ln L(\alpha, \xi_1, \xi_2) = \sum_{i=1}^{N} n_i \ln \nu_i(\alpha, \xi_1, \xi_2) \ , \tag{2}$$

where n_i is the number of entries in bin i, with $i = 1, \ldots, N$, and ν_i is the predicted of entries. This is found from

$$\begin{aligned} \nu_i(\alpha, \xi_1, \xi_2) &= n_{\rm tot} \int_{x_{{\rm min},i}}^{x_{{\rm max},i}} f(x; \alpha, \xi_1, \xi_2)\, dx \\ &\approx n_{\rm tot} f(x_i; \alpha, \xi_1, \xi_2) \Delta x \ , \end{aligned} \tag{3}$$

where $n_{\rm tot} = \sum_{i=1}^{N} n_i$, $x_i = (x_{{\rm min},i} + x_{{\rm max},i})/2$ is the middle of the ith bin and $\Delta x = x_{{\rm max},i} - x_{{\rm min},i}$ is the bin width. The approximation in the second line of (3) valid as long as the pdf is roughly linear across the bin.

To read in the histogram and access the bin contents you can use code of the form:

```
TFile* histFile = new TFile (fileName.c_str(), "READ");
TH1D*  h = (TH1D*)histFile->Get(histName.c_str());
int nBins = h->GetNbinsX();
vector<double> y(nBins);
for (int i=0; i<nBins; i++){
  y[i] = h->GetBinContent(i+1);
}
```

Note that the contents of the histogram are accessed using elements 1 to `nBins`, whereas the vector `y` goes from 0 to `nBins-1`.