Wintersemester 2012/2013, ESS

Übung: Measurement error propagation

Assignment due date: 14.1.2013

This exercise deals with measurement error propagation when calculating values derived from measurements. The measurement errors are assumed to follow a normal distribution. Here, you will calculate the effects of error propagation using an analytical approximation as well as a numerical simulation.

Rescaling and shifting the normal distribution

We will need samples from a normal distribution. You can use one of the methods developed in the previous exercise or use a function from a standard library. In case you are using your own or someone else's code that produces normally distributed numbers with mean 0 and variance 1, you can transform them to samples from an arbitrary normal distribution.

Samples g from a normal distribution with mean 0 and variance 1 can be rescaled by

$$g_{\mu,\sigma} = \mu + \sigma g$$

to give normally distributed numbers $g_{\mu,\sigma}$ with mean μ and standard deviation σ.

Analytical approximation for error propagation

The mean of the distribution of $f(x, y, ...)$ is assumed to be f applied to the mean values $\mu_x, \mu_y, ...$ of the input distributions:

$$\mu_f \approx f(\mu_x, \mu_y, ...)$$

As mentioned in the lectures, we can approximate the variance σ_f^2 of $f(x, y, ...)$ as

$$\sigma_f^2 \approx \left(\frac{\partial f}{\partial x}(\mu_x, \mu_y, ...)\right)^2 \sigma_x^2 + \left(\frac{\partial f}{\partial y}(\mu_x, \mu_y, ...)\right)^2 \sigma_y^2 + ...$$

The term $\frac{\partial f}{\partial x}(\mu_x, \mu_y, ...)$ means the following: first, you have to form the partial derivative $\frac{\partial f}{\partial x}$. This partial derivative again is a function of $x, y, ...$ and it is evaluated at the point $(\mu_x, \mu_y, ...)$ in the formula.

Monte Carlo integration

If we do not have a reliable analytical formula to calculate the effects of error propagation, we can estimate the result numerically by drawing N samples from the distribution of the input variables $x, y, ...$ and calculating the resulting $f(x, y, ...)$.

If the values calculated with f are named $s_1, s_2, ..., s_N$, then the estimate $\hat{\mu_f}$ for the mean value of f is given by the arithmetic mean

$$\hat{\mu_f} = \frac{1}{N} \sum_{i=1}^{N} s_i$$

and the estimate of the variance $\hat{\sigma_f^2}$ is

$$\hat{\sigma_f^2} = \frac{1}{N-1} \sum_{i=1}^{N} (s_i - \hat{\mu_f})^2$$

As N goes to infinity, the estimates $\hat{\mu_f}$ and $\hat{\sigma_f^2}$ will converge towards the true values μ_f and σ_f^2.

Plotting histograms

You may use any method you like to plot histograms. The easiest way is to reuse the R script from the previous exercise that can be found under `/home/matthies/uebung-rng/plot.r`.

Your assignment

Please submit a brief report with your answers to ast_uebung[at]zbh[dot]uni-hamburg[dot]de (please include your full name). Attach one file containing the answers to all assignments (only .txt or .pdf) as well as your source code and instructions on how to compile and run it (preferably a Makefile).

Aufgabe 1: Area of a rectangle

The area A of a rectangle with side lengths x, y is given by $A = xy$. The side lengths have been measured with means μ_x, μ_y and standard deviations of σ_x, σ_y.

What is σ_A as given by the analytical approximation?

Program a Monte Carlo simulation to estimate σ_A numerically. Your program should take the following command-line arguments:

```
./area <mode> N mu_x sigma_x mu_y sigma_y seed
```

where `<mode>` is either "sample" or "compare". In "sample" mode, your program should simply print out N samples for the calculated area. This output can then be plotted in a histogram using the aforementioned R script (or any other method if you prefer). In "compare" mode, your program should only compare the mean and standard deviation of the area as calculated via Monte Carlo integration to the analytical approximation.

Choose values for $\mu_x, \sigma_x, \mu_y, \sigma_y$ and plot a histogram of the calculated areas. What does the distribution look like? How does the distribution and calculated μ_A, σ_A change when you increase N?

Compare the calculated values for μ_A, σ_A with the values given by the analytical approximation. Do they agree?

Aufgabe 2: Equilibrium constant of dimerisation

Suppose a protein A can bind to itself to form the dimer D. The reaction is written as $2A \rightleftharpoons D$, and the equilibrium constant k is given by

$$k = \frac{[A]^2}{[D]}$$

where $[A]$ is the concentration of the protein and $[D]$ is the concentration of the dimer. Assuming the measurements of $[A]$ and $[D]$ have means μ_A, μ_D and standard deviations σ_A, σ_D, we want to find the resulting errors in k.

Derive the analytical approximation for σ_k and write a program that performs a Monte Carlo simulation like in the previous task. Your program should take the following command-line arguments:

```
./dimer <mode> N mu_A sigma_A mu_D sigma_D seed
```

where `<mode>` should work like in the previous task.

What does the histogram for k look like? Are there values for $\mu_A, \mu_D, \sigma_A, \sigma_D$ where the shape of the histogram looks qualitatively different? Can you imagine a reason why?

Compare your numerical results with the analytical approximation. Can you find values for $\mu_A, \mu_D, \sigma_A, \sigma_D$ where they disagree even if you make N very large? Which result do you believe and why?