Answers

Timm Ruland & Boris Prochnau

April 20, 2014

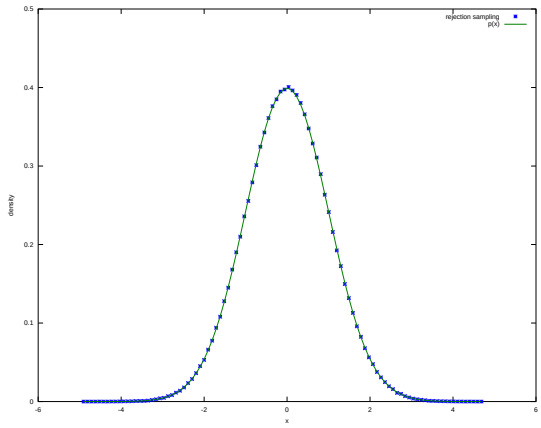
1 Task: 1

- What does it do?
  it prints two not sufficiently seeded "random" variables
- The difference between the methods is that rand() returns a random number and the other method
  - rand() returns a random integer between 0 and RANDMAX
  - gsl_rng_mt19937 is a generator that generates random numbers

  In the code the generator is passed to a (Distribution)function that uses this generator to evaluate a random number depending on the Distribution specified in the function.
- What happens if you remove the expression (double)?
  The division operation $\frac{rand()}{RANDMAX}$ does $\frac{intSmall}{intBig}$ and should result in a double, but with no cast it will be floored to 0.
- Is there a direct function to generate normally distributed random variables?
  double gsl_ran_gaussian_pdf(const gsl_rng * r, double sigma) command

2 Task: 3

The Plot should simulate the occurrence density of points of our 1 000 000 samples which are normally distributed. But the picture shows that all points are over the exact plot of the density function what is nearly impossible with so many samples. Therefore we conclude that the plot was made with different parameters.



3 Task: 4

We can sample a uniformlly y in the intervall [0,1] and apply the inverse cdf to sample a normally distributed variable

4 Task: 5

The Algorithm uses a interpolation (see fig. 1) with maximal estimation error: $2 \cdot 10^{-9}$ (see fig. 2) where the upper half of the distribution function is divided into 2 sections in which interpolations where made. If x is in the lower half of the distribution function, than 1-NomralCDF(-x) will be used to use the symmetric property.

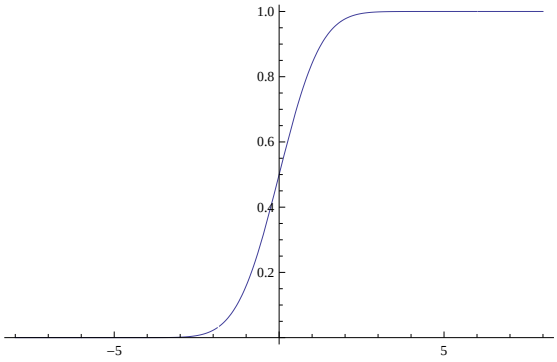


Figure 1: Plot of Approximation

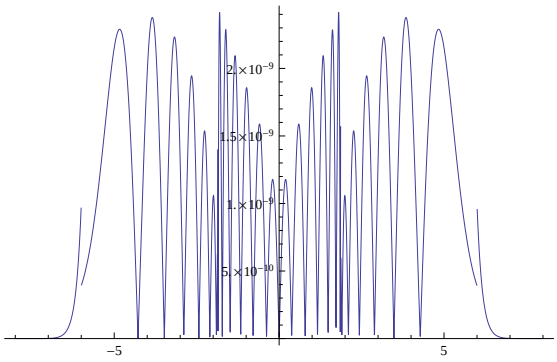


Figure 2: estimation error

5 Task 6

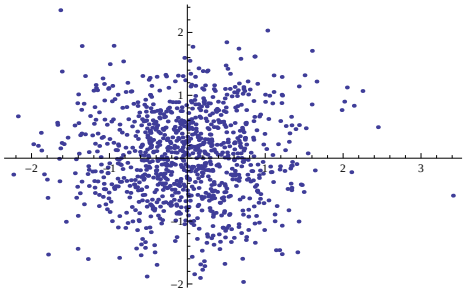


Figure 3: 2D Plot

6 Task 7

Let X, Y be two independent standard normally distributed random variables. The joint distribution of them is

$$p(x,y) = p(x)p(y) = \frac{1}{\sqrt{(2\pi)}} e^{-\frac{x^2}{2}} \cdot \frac{1}{\sqrt{(2\pi)}} e^{-\frac{y^2}{2}} = \frac{1}{2\pi} e^{-\frac{y^2+x^2}{2}}$$

We notice that with x,y being two Cartesian coordinates: $x^2 + y^2$ is equal to their radius r^2 on a polar circle and we can write:

$$p(x,y) = \frac{1}{2\pi} e^{-\frac{r^2}{2}}$$

which is the product of two density functions:

- r^2 $\quad \text{Exp}(\frac{1}{2})$ so $\lambda = 1/2$.
- θ $\quad \text{Unif}(0, 2\pi)$

By using the method of the inverse cdf before, we can now make some interesting computations:

$$\begin{aligned} y &= 1 - \exp(-\lambda x) && | -1 \\ y - 1 &= -\exp(-\lambda x) && | \cdot (-1) | \log \\ log(1-y) &= -\lambda x && | : (-\lambda) \\ \frac{-log(1-y)}{\lambda} &= x \end{aligned}$$

Therefore we know that the inverse of the exponential distribution function is $F^{-1}(y) = \frac{-\log(1-y)}{\lambda}$. And now we can sample exponentially distributed r.v.'s with the inverse cdf method used before:

$$Exp(\lambda) = \frac{-\log(1-y)}{\lambda}$$

and more important:

$$r \sqrt{-\log(Unif(0,1))}$$

with this knowledge we can now sample two uniform distributed r.v.'s (u_1, u_2), one for r and the other for θ:

- $r \sqrt{-\log(u_1)}$
- $\theta = 2\pi u_2$

But those are polar coordinates, so we need to transform them to Cartesian coordinates first:

- $x = r\cos(\theta)$
- $y = r\sin(\theta)$

with these we now got back to a sample from the joint gaussian r.v. what explains why they are normally distributed (with the first equation above).

7 Task 8

The algorithm need only one summation instead of two. The algorithm interates new samples to the total solution of the last step, so it could work better if we constantly give new sampels to the old ones because there is no need to run the hole calculation again.

8 Task 9

We think that the error drops with the rate $\frac{1}{\sqrt{N}}$.

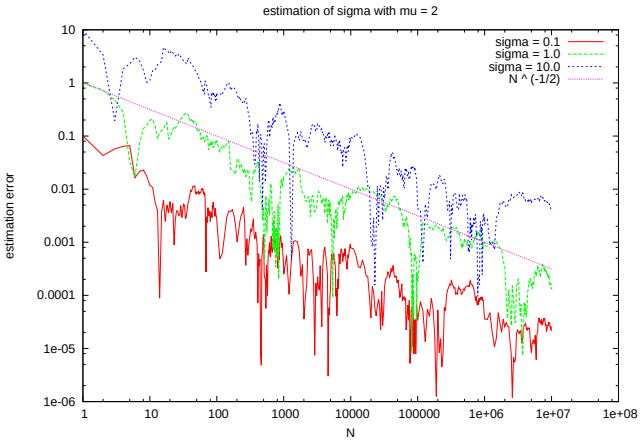


Figure 4: estimation error