

Tutorial 6 - Importance Sampling

Computer Graphics

Kartic Subr and Martin Asenov
November 11, 2019

Don't look inside `function.cpp` as this will give you a strong hint for the solution!

You have been given a task to come with a better method for sampling a 'secret' function. The 'secret' function can be seen in fig.1,(first column), named by $f(x)$.

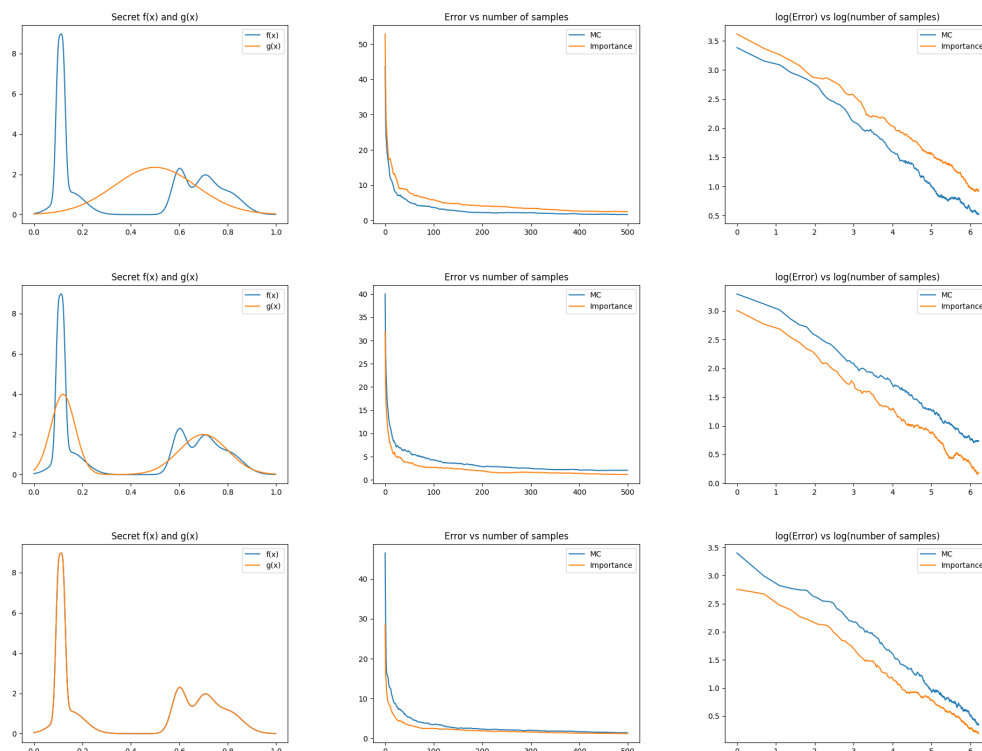


Figure 1: Importance sampling. (first column) PDFs of $f(x)$, the secret function, and $g(x)$ used for Importance Sampling (second column) Convergence of the ground truth mean with number of samples (third column) Log-error vs Log-number of samples. Note that due to some limitations of using matplotlib with C++, the $\log(\text{error})$ and $\log(\text{number of samples})$ were plotted, rather than changing the scale of the plots.

You have been provided `sample.cpp` which all of the code to get you started. Inside the code you will find implementation of Monte Carlo sampling and Importance Sampling. For importance sampling a Mixture of Gaussian has been implemented for $g(x)$. Your task is to figure out the

best parameters for $g(x)$ - μ and σ for how many gaussians you want to use. You can compile `sample.cpp` on a DICE machine using the command:

```
g++ sample.cpp function.cpp -I/usr/include/python2.7 -lpython2.7 -o sample
```

You can use any number of gaussians - in the solution visualized in fig.1, only 1, 2 and 6 gaussians were used respectively, with the 6 gaussian fitting the distribution exactly. The compile command assumes you have python2.7 installed with matplotlib and numpy (for plotting purposes, it should work out of the box on DICE machines).

When you run `./sample` this will produce `pdf.png`, `error.png` and `log_error.png`. Depending on the quality of the fit, you should see results similar as in fig.1.