Abstract

Morals About (Artificially) Injecting Pseudorandomness in Computations of Deterministic Quantities

Computations in computer simulations, engineering and machine learning are of a numerical kind. They are trying to compute a quantity that doesn't have a closed form expression but which can only be approximated through repeated computations. Numerical methods solve integration, optimization and linear algebra problems. I focus on the first situation in which we have an integral to evaluate. I ask the question: What is a good numerical method to approximately compute integrals? Each method comes with its accuracy and efficiency profile which is used to figure out whether it is a good approach to follow. In contrast to classical ‚deterministic' numerical integration methods, there are stochastic Monte-Carlo methods: You input a domain, a function, and it returns a calculated value of a definite integral. So, input and output is identical to with the standard numerical integration methods. But under the hood it's quite different. They do not consist of a geometric-analytic decomposition of the problem, but rather of a certain sampling method of the integrand which is based on the use of (pseudo-) random numbers. The use Monte-Carlo methods are mostly pragmatically justified: they do not suffer from the curse of dimensionality and provide an excellent tradeoff between the important features of computational methods, esp. efficiency and accuracy.

Formal statements of the properties of stochastic algorithms, e.g. unbiasedness of the point estimator, often depend on the use of "real" random numbers. I want to raise some doubts on the utility of and the mathematical justification for the use of pseudo-random number generators (PRNG) in computation.
The purpose of a PRNG is to render its outcome an *unpredictable* function of the seed. Thus, the unpredictability of these numbers rest completely on the user not knowing the random number generator used in the computation (and its seed). Hence, to save the pseudorandomness-status, one actively chooses not to know the seed. It sounds odd that this requires the hiding of information.
It reveals that pseudo-randomness is then very much dependent on prior subjective knowledge.

If randomness is our crutch to describe a lack of predictability, then *uncertainty*, formalized by probability theory, provides a much cleaner mathematical concept, which does not require the ability to draw "random numbers" at all. In fact, many classical numerical algorithms can be interpreted in a bayesian way as agents manipulating probability distributions (not random numbers) from priors to posteriors, with an associated notion of uncertainty. Thus, I conclude that uncertainty-guarantees from pseudo-randomness are ill-suited for the rule of numerical computation.