MA5233 Computational Mathematics

Lecture 24: Monte Carlo

Simon Etter

2019/2020

Monte Carlo

Introductory example

Assume we want to compute the integral

$$I := \int_0^1 \dots \int_0^1 f(y_1, \dots, y_d)\, dy_1 \, \dots \, dy_d.$$

Applying a one-dimensional quadrature rule $(x_k, w_k)_{k=1}^n$ with error $e_n = \mathcal{O}\big(n^{-p}\big)$ repeatedly, we obtain

$$\begin{aligned} I &= \int_0^1 \dots \int_0^1 \left(\sum_{k_1=1}^{n} f(x_{k_1}, y_2, \dots, y_d)\, w_{k_1} + \mathcal{O}\big(n^{-p}\big) \right) dy_2 \, \dots \, dy_d \\ &= \dots \\ &= \sum_{k_d=1}^{n} \dots \sum_{k_1=1}^{n} f(x_{k_1}, \dots, x_{k_d})\, w_{k_1} \, \dots \, w_{k_d} + \mathcal{O}\big(n^{-p}\big). \end{aligned}$$

Observation: for increasing d, number of function evaluations N must scale as $\mathcal{O}\big(n^d\big)$ to achieve a constant error $\mathcal{O}\big(n^{-p}\big) = \mathcal{O}\big(N^{-p/d}\big)$!

This phenomenon has been called the *curse of dimensionality*.

Monte Carlo

Introductory example (continued)

Different approach: reinterpret I as the expextation value

$$I = \mathbb{E}[f(X_1, \ldots, X_d)], \qquad X_k \overset{\text{iid}}{\sim} \text{Uniform}[0, 1].$$

Expectations can be computed by taking the average of a sufficiently large number of samples $x_k^{(i)}$,

$$\mathbb{E}[f(X_1, \ldots, X_d)] = \frac{1}{N} \sum_{i=1}^{N} f\left(x_1^{(i)}, \ldots, x_d^{(i)}\right) + \mathcal{O}\left(N^{-1/2}\right).$$

$\mathcal{O}\left(N^{-1/2}\right)$ error term follows from basic statistics.
We will discuss this later.

Observation: N function evaluations lead to $\mathcal{O}\left(N^{-1/2}\right)$ error!

In particular, Monte Carlo is better than midpoint rule ($p = 2$) for $d > 4$.

See `convergence()`.

Monte Carlo

Abstract Monte Carlo

$$\mathbb{E}[F] \approx Q := \frac{1}{N}\sum_{k=1}^{N} F_k \qquad \text{where} \qquad F, F_k \overset{\text{iid}}{\sim} \mathcal{F}.$$

Many things can be written as expectations:

- Sums and integrals (see introductory example).
- Probabilities: $P(X \in S) = \mathbb{E}[f(X)]$ where $f(X) = \begin{cases} 1 & \text{if } X \in S, \\ 0 & \text{otherwise.} \end{cases}$

Another example

- Assume we are playing a game like chess or go and we are unsure which move to make.
- Clearly, a move is good if it increases our chances of winning.
- This probability can be estimated by playing the game several times with random moves and counting how many times we won.
- This is one of the tools that AlphaGo used to beat humans in go.

Monte Carlo

Further remarks

- Monte Carlo idea: replace deterministic but lengthy calculations with random sampling.
- This idea was used for the first time by the physicists who developed nuclear weapons for the US army.
- "Monte Carlo" was originally a code name in reference to a casino of the same name.

Error estimation

- Note that the Monte Carlo estimate $Q \approx \mathbb{E}[F]$ is a random variable.
- For any N, it is thus possible that Q is a bad approximation to $\mathbb{E}[F]$. Conversely, even $N = 1$ may yield the exact answer.
- The power of Monte Carlo methods is that the event $(\left|Q - \mathbb{E}[F]\right| > C)$ becomes increasingly unlikely for $N \to \infty$.
- Central limit theorem on next slide provides a rigorous statement of this idea.

Monte Carlo

Central limit theorem

Assume $F, F_1, \ldots, F_N \overset{\text{iid}}{\sim} \mathcal{F}$. Then, $Q := \frac{1}{N} \sum_{k=1}^{N} F_k$ is approximately normally distributed with $\mathbb{E}[Q] = \mathbb{E}[F]$ and $\mathrm{Var}[Q] = \frac{\mathrm{Var}[F]}{N}$.

Recap

- $\mathrm{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$.
- $X \sim \mathcal{N}(\mu, \sigma^2) :\iff P(X \in \mathcal{X}) \propto \int_{\mathcal{X}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx$
  Also, $X \sim \mathcal{N}(\mu, \sigma^2) \implies \mathbb{E}[X] = \mu$ and $\mathrm{Var}[X] = \sigma^2$.

Application to Monte Carlo

- Expected error: $\sqrt{\mathbb{E}[(Q - \mathbb{E}[F])^2]} = \sqrt{\frac{\mathrm{Var}[F]}{N}}$.
- Confidence interval: $P\left(\left|Q - \mathbb{E}[F]\right| \leq C(\varepsilon) \sqrt{\frac{\mathrm{Var}[F]}{N}}\right) = 1 - \varepsilon$.

See `variance_estimation()`.

Conclusions

- $\mathcal{O}\left(N^{-1/2}\right)$ convergence in N.
- Prefactor determined by $\mathrm{Var}[F]$.

Monte Carlo

Simulation of random variables

Two stages:

- Generate $U \sim \text{Uniform}[0, 1]$.
- Map $X = f(U)$ such that X has the desired distribution.

Def: Pseudo-random number generator (pRNG)

Deterministic algorithm mapping some seed s to a sequence $U_k \in [0, 1]$ which looks as if $U_k \overset{\text{iid}}{\sim} \text{Uniform}[0, 1]$.

Remarks:

- There is no rigorous definition of "looks as if $U_k \overset{\text{iid}}{\sim} \text{Uniform}[0, 1]$". In practice, pRNGs are assessed by measuring their performance on a large number of statistical tests.

  See e.g. `https://en.wikipedia.org/wiki/Diehard_tests`.
- pRNGs have at least two advantages over "true" RNGs:
  - They are faster, see `rng_benchmarks()`.
  - They allow to exactly reproduce numerical results, which is useful for testing and debugging.

Monte Carlo

Thm: Transformation

Let F be a cumulative distribution function on $\mathbb{R}$, i.e. $F : \mathbb{R} \to [0, 1]$ is non-decreasing, right-continuous and $F(-\infty) = 0$, $F(\infty) = 1$.

Let $U \sim \text{Uniform}[0, 1]$.

Then, $X := F^{-1}(U) \sim F$.

Proof. $P\big(X \leq x\big) = P\big(F^{-1}(U) \leq x\big) = P\big(U \leq F(x)\big) = F(x)$.

Example

Consider triangular density function $f(x) = 2x$ on $[0, 1]$.

We have $F(x) = \int_0^x 2x' \, dx' = x^2$; hence $X := \sqrt{U} \sim f$.

See `transformation_sampling()`.

Discussion

Good: sampling by transformation is very efficient if $F^{-1}(U)$ is cheap.

Bad: only one-dimensional random variable have cumulative distribution function, and computing $F^{-1}(U)$ can be difficult.

Monte Carlo

Thm: Rejection sampling

Let $f(x), g(x)$ be probability densities on Ω, i.e. $f, g : \Omega \to \mathbb{R}$ such that $\int_\Omega f(x)\,dx = \int_\Omega g(x)\,dx = 1$. Assume further that $f(x) \le C\,g(x)$.
Consider the samples F generated by the following algorithm.

1. Sample $G \sim g$ and $U \sim \text{Uniform}[0, 1]$.
2. If $U \le \frac{f(G)}{C\,g(G)}$, then return $F := G$.
   Otherwise, repeat from step 1.

We have $F \sim f$.

Proof.

$$
\begin{aligned}
P(F \in \mathcal{X}) &= P(G \in \mathcal{X} \mid G \text{ is accepted}) \\
&= \frac{P(G \in \mathcal{X} \text{ and } G \text{ is accepted})}{P(G \text{ is accepted})} \\
&= \frac{\int_{\mathcal{X}} g(x)\,\frac{f(x)}{C\,g(x)}\,dx}{\int_\Omega g(x)\,\frac{f(x)}{C\,g(x)}\,dx} = \int_{\mathcal{X}} f(x)\,dx.
\end{aligned}
$$

Monte Carlo

Discussion

Good: rejection sampling works for almost all f.

Bad: rejection sampling can be very inefficient if $g(x)$ is a bad approximation to $f(x)$.

More precisely, we have $P(G \text{ is accepted}) = \frac{1}{C}$, i.e. on average we generate C samples of G and U for every sample of F.

Example

Consider $f(x) := 2x$ and $g(x) := 1$ on $\Omega = [0, 1]$.

We have $f(x) \leq 2g(x)$; hence rejection sampling discards half of the samples.

See `rejection_sampling()`.

Monte Carlo

Thm: Importance sampling

Let f, g be probability densities on Ω and $F \sim f$, $G \sim g$.
Let h be a function on Ω. Then,

$$\mathbb{E}[h(F)] = \mathbb{E}[h(G)\,\tfrac{f(G)}{g(G)}]$$

Proof.

$$\mathbb{E}[h(F)] = \int_\Omega h(x)\,f(x)\,dx = \int_\Omega h(x)\,\frac{f(x)}{g(x)}\,g(x)\,dx = \mathbb{E}[h(G)\,\tfrac{f(G)}{g(G)}].$$

Discussion

Importance sampling theorem is useful if:

- G is easier to sample than F.
- $h(G)\,\frac{f(G)}{g(G)}$ has lower variance than $h(F)$.

Monte Carlo

Example

Consider $f(x) := 1$ and $g(x) := 2x$ on $\Omega = [0, 1]$, and $h(x) = 2x$.

We observe:

$$h(F) = 2F \quad \Longrightarrow \quad \mathrm{Var}[h(F)] = \tfrac{1}{3},$$
$$h(G)\,\tfrac{f(G)}{g(G)} = 1 \quad \Longrightarrow \quad \mathrm{Var}[h(G)\,\tfrac{f(G)}{g(G)}] = 0.$$

Monte Carlo applied to $\mathbb{E}[h(G)\,\frac{f(G)}{g(G)}]$ is exact!

See `importance_sampling()` for a more realistic example.

Monte Carlo

Review Monte Carlo

- $\mathcal{O}\left(N^{-1/2}\right)$ convergence of Monte Carlo is very slow, but it is faster than more traditional algorithms in high dimensions.
- Unlike most algorithms we have seen, Monte Carlo is straightforward to parallelise and runs efficiently on even the largest machines.

  See `24pp_parallel_monte_carlo.jl`, `https://www.karlrupp.net/wp-content/uploads/2015/06/40-years-processor-trend.png` and `https://www.top500.org/lists/2019/06/`.
- Monte Carlo also suffers from curse of dimensionality if $\mathrm{Var}[F] \propto a^d$.

  Monte Carlo can be effective in some cases because it implicitly exploits structure in the problem.

References and further reading

Excellent lecture notes:

- `https://warwick.ac.uk/fac/sci/statistics/staff/academic-research/johansen/teaching/mcm-2007.pdf`
- `https://statweb.stanford.edu/~owen/mc/`