MA3227 Numerical Analysis II

Lecture 08: Monte Carlo Methods

Simon Etter

Semester II, AY 2020/2021

Monte Carlo Methods

Problem statement

Estimate the expectation $\mathbb{E}[X]$ of a random variable X by computing the average of a large number of samples of X.

This problem statement raises several questions.

- *Why compute expectations?*

  According to basic probability theory, the expectation $\mathbb{E}[X]$ of a random variable X which assumes values x in a discrete or continuous set $\mathcal{X}$ with probability $p(x)$ is given by, respectively,

  $$\mathbb{E}[X] = \sum_{x \in \mathcal{X}} x\, p(x) \qquad \text{or} \qquad \mathbb{E}[X] = \int_{\mathcal{X}} x\, p(x)\, dx.$$

  Computing an expectation is hence the same as evaluating a particular sum or integral, and conversely any sum or integral can be reinterpreted as an expectation.

- *Why compute expectations through sampling?*

  This question is best answered by means of some examples; see the following slides.

Monte Carlo Methods

Example 1: Winning probabilities in m, n, k-games

Consider the m, n, k-*game* described under

`https://en.wikipedia.org/wiki/m,n,k-game`.

Assume we want to compute the probability P of a win for player 1 assuming random moves on behalf of both players.

This probability could be computed as the sum

$$P = \sum_{\text{all possible games}} [\text{ probability of game }] \times \begin{cases} 1 & \text{if player 1 wins,} \\ 0 & \text{otherwise,} \end{cases}$$

but this sum contains roughly $(mn)!$ terms and therefore cannot be evaluated except for very modest values of m and n.

For example, if we assume a runtime of just 1 nanosecond per game, then evaluating the above sum for $m = n = 4$ would take about 6 hours, and evaluating the sum for $m = n = 5$ would take about 500'000 years!

Monte Carlo Methods

Example 1: Winning probabilities in m, n, k-games (continued)

These ludicrous runtimes can be avoided if we rewrite the above sum as the expectation

$$P \approx \mathbb{E}[X] \qquad \text{where} \qquad X = \begin{cases} 1 & \text{if player 1 wins,} \\ 0 & \text{otherwise} \end{cases}$$

and then estimate this expectation as follows.

- Play out N random games.
- For each such game i, record in the variable $X_i \in \{0, 1\}$ whether player 1 won.
- Estimate $\mathbb{E}[X] \approx \frac{1}{N} \sum_{i=1}^{N} X_i$.

This approach is known as *Monte Carlo sampling*, and `mnk_probabilities()` shows that it leads to reasonably accurate estimates already for moderate values of N.

Monte Carlo Methods

Example 2: High-dimensional integrals

Assume we want to compute a d-dimensional integral

$$I = \int_0^1 \dots \int_0^1 f(x_1, \dots, x_d)\, dx_1 \,\dots\, dx_d.$$

Perhaps the most obvious way to approximate this quantity is to apply a one-dimensional quadrature rule $(x_k, w_k)_{k=1}^n$ to each of these d integrals, i.e. to compute

$$I \approx \sum_{k_d=1}^{n} \dots \sum_{k_1=1}^{n} f(x_{k_1}, \dots, x_{k_d})\, w_{k_1} \,\dots\, w_{k_d}.$$

We observe:

- The above approximation requires $N = n^d$ function evaluations.
- If the one-dimensional quadrature rule has an $O(n^{-p})$ error, then so does the high-dimensional quadrature approximation.

  Proof. See next slide.

Monte Carlo Methods

Proof that high-dimensional quadrature inherits the order of convergence of the one-dimensional quadrature rule.

Iteratively replacing integrals with quadrature approximations, we obtain

$$\begin{aligned}
&\int_0^1 \dots \int_0^1 f(x_1, \dots, x_d)\, dx_1 \, \dots \, dx_d = \dots \\
&\quad = \int_0^1 \dots \int_0^1 \left(\sum_{k_1=1}^{n} f(x_{k_1}, x_2, \dots, x_d)\, w_{k_1} + O\big(n^{-p}\big) \right) dx_2 \, \dots \, dx_d \\
&\quad = \quad \dots \\
&\quad = \sum_{k_d=1}^{n} \dots \sum_{k_1=1}^{n} f(x_{k_1}, \dots, x_{k_d})\, w_{k_1} \, \dots \, w_{k_d} + O\big(n^{-p}\big).
\end{aligned}$$

Example 2: High-dimensional integrals (continued)

Our observations on slide 5 imply that the error and number of function evaluations $N = n^d$ are related through

$$\text{error} \;=\; O\big(N^{-p/d}\big);$$

see `integral_via_quadrature()`.

The number of function evaluations required to meet a certain error tolerance is hence given by

$$N \;=\; O\big(\text{error}^{-d/p}\big),$$

i.e. N scales exponentially in the number of dimensions d and therefore becomes prohibitively large already for moderate values of d.

This phenomenon is known as the "curse of dimensionality".

Monte Carlo Methods

Example 2: High-dimensional integrals (continued)

As in the m, n, k-game example, it turns out that we can avoid excessive runtimes by replacing deterministic computations with random sampling.

In the case of high-dimensional integrals, the key to doing so is to observe that we can interpret such integrals as the expectation

$$\mathbb{E}[f(X)] = \int_0^1 \dots \int_0^1 f(x_1, \dots, x_d)\, dx_1 \, \dots \, dx_d$$

and hence

$$\mathbb{E}[f(X)] \approx \frac{1}{N} \sum_{k=1}^{N} f(X_{k,1}, \dots, X_{k,d})$$

where

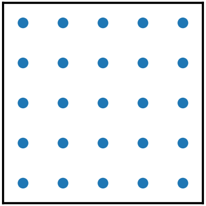
- X is a random variable uniformly distributed over $[0,1]^d$, and
- $X_k \in [0,1]^d$ is a sequence of samples of this random variable.

This trick is again known as *Monte Carlo sampling*.

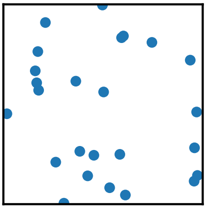
Monte Carlo Methods

Example 2: High-dimensional integrals (continued)

Illustration of quadrature vs. Monte Carlo sampling:



Quadrature



Monte Carlo

Blue dots denote quadrature / sampling points.

Monte Carlo Methods

Example 2: High-dimensional integrals (continued)

`integral_via_monte_carlo()` shows that Monte Carlo sampling converges with rate $O(N^{-1/2})$ regardless of the dimension.

We therefore conclude:

- Monte Carlo sampling is slower than midpoint quadrature if

$$2/d > 1/2 \quad \Longleftrightarrow \quad d < 4.$$

- Monte Carlo sampling is faster than midpoint quadrature if

$$2/d < 1/2 \quad \Longleftrightarrow \quad d > 4.$$

In particular, Monte Carlo sampling avoids the curse of dimensionality and is therefore a powerful tool to tackle high-dimensional problems.

Example 1: Winning probabilities in m, n, k-games (reprise)

It turns out that now that we know what to look out for, we can also observe the $O(N^{-1/2})$ convergence behaviour when applying Monte Carlo sampling to compute the m, n, k-winning probabilities: every time we run `mnk_probabilities()` with a 100x larger number of samples N, we gain roughly one extra digit of accuracy.

Introduction to Monte Carlo error estimation

The above examples indicate that the power of the Monte Carlo approach stems from the fact that

$$\mathbb{E}[X] \;=\; \tfrac{1}{N}\sum_{k=1}^{N} X_k + O(N^{-1/2})$$

for a very wide class of random variables X.

Our goal in the following will be to clarify and prove this claim. Doing so requires a solid background in probability theory; hence let us begin by recapitulating the basics.

Monte Carlo Methods

Probability theory in a nutshell

- A *measure* is a function P which maps subsets of some set Ω to nonnegative real numbers such that

$$A, B \subset \Omega \quad \text{and} \quad A \cap B = \{\} \quad \Longrightarrow \quad P(A \cup B) = P(A) + P(B).$$

- A *probability measure* is a measure P such that $P(\Omega) = 1$.
- A *probability space* is a pair (Ω, P) such that P is a probability measure on the set Ω.
- A *random variable* is a function $X : \Omega \to \mathcal{X}$ defined on a probability space (Ω, P).
- The probability measure $\hat{P}$ on $\mathcal{X}$ given by

$$\hat{P}(A) \;=\; P(\{\omega \in \Omega \mid X(\omega) \in A\})$$

  is called the *distribution* of the random variable $X : \Omega \to \mathcal{X}$.
- Probabilities associated with random variables are commonly expressed using the following convenience notations:

$$P(X \in A) = P\big(\{\omega \in \Omega \mid X(\omega) \in A\}\big),$$
$$P(X_1 \in A_1, X_2 \in A_2) = P\big(\{\omega \in \Omega \mid X_1(\omega) \in A_1 \wedge X_2(\omega) \in A_2\}\big).$$

Monte Carlo Methods

Example 1: Winning probabilities in m, n, k-games (continued)

The above definitions were fairly abstract, so let me illustrate them by means of the m, n, k-games example from the beginning of this lecture.

This example can be mapped to the above framework as follows.

- The probability space Ω is the set of all sequences of board states

  $$\omega = B_0 \to B_1 \to \ldots \to B_g$$

  which represent a rules-conforming and complete game.

- Since this set is discrete, we can specify the probability measure P by specifying the probability $P(\{\omega\})$ of each sequence $\omega \in \Omega$ and then set

  $$P(A) = \sum_{\omega \in A} P(\{\omega\}) \qquad \text{for all } A \subset \Omega.$$

Monte Carlo Methods

Example 1: Winning probabilities in m, n, k-games (continued)

- According to the "random moves" assumption, the probability of a game of length g is given by

$$P\big(\{B_0 \to \ldots \to B_g\}\big) \;=\; \prod_{i=0}^{g}\big(mn - i\big)^{-1}.$$

- The winning probability for player 1 is then given by $P(W_1 = 1)$ where W_1 denotes the random variable $W_1 : \Omega \to \{0, 1\}$ given by

$$W_1\big(B_0 \to \ldots \to B_g\big) \;=\; \begin{cases} 1 & \text{if } B_g \text{ shows a win for player 1, and} \\ 0 & \text{otherwise.} \end{cases}$$

Monte Carlo Methods

[To be continued]