

Моделирование, метод Монте-Карло. Короткий обзор

Нина Эдуардовна Голяндина
к.ф.-м.н., доцент кафедры статистического моделирования

Санкт-Петербургский государственный университет
03 декабря 2019, Санкт-Петербург

- Многие события случайные и часто нужно имитировать их поведение, чтобы на основе многих случайных имитаций понять, какие характеристики могут быть в результате. Например, риск проиграть, время простоя в порту, ...
- Есть другой класс задач, когда нужно посчитать что-то неслучайное, но аналитически результат не получить. В этом случае применяются численные методы. Если задача сложная или если нужно простое решение, то применяется статистическое моделирование
- В любом случае, нужен некий источник случайности. Он называется Датчик псевдослучайных чисел. Метод моделирование — это метод, который на вход получает то, что генерирует датчик, а на выходе дает результат с нужным распределением.

Генератор псевдослучайных чисел

- Датчик (генератор) псевдослучайных чисел генерирует равномерно распределенные на $[0, 1]$ числа. Т.е. если вам нужно независимо бросать на $[0, 1]$ точки, то можно воспользоваться таким датчиком.
- Последовательность чисел такая, что очень похожа на независимую повторную выборку их р.р. на $[0, 1]$. Соответственно, качество датчика измеряется тем, насколько похожа.
- Последовательность не случайная. Т.е. если задать конкретное начальное значение (`set.seed(1)` в R), будете получать одну и ту же последовательность. Более того, последовательность состоит из целых чисел на $[0, 2^d - 1]$, где d — разрядность системы (например, 64), которые потом делятся на 2^d , чтобы получилось вещественное число на $[0, 1]$.
- Так как исходно строится последовательность целых чисел, где следующее число строится на основе предыдущих, то последовательность неминуемо с какого-то момента начнет повторяться. Поэтому период (число значений без повторения) тоже является характеристикой датчика.

- Один из лучших современных генераторов — ‘Вихрь Мерсенна’
https://en.wikipedia.org/wiki/Mersenne_Twister
Обычно именно он используется в пакетах для генерации.
- Самый простой датчик случайных чисел — мультипликативный генератор:
 $U_{i+1} = M \cdot U_i \bmod (2^d)$. Здесь M — параметр датчика, мультипликатор, т.е. предыдущее значение умножается на M и берется остаток от деления на 2^d , где d — размерность системы. Результат работы датчика — $\alpha_i = U_i/2^d$.
- Понятно, что U_0 задает все дальнейшее поведение датчика. Оно должно быть нечетное.
- Пример плохого мультипликатора — 2. Пример хорошего — 663608941 (для разрядности 32).

Что такое моделирование заданного распределения?

- Алгоритм моделирования на основе обращений к датчику псевдослучайных чисел выдает результаты с необходимыми свойствами. Для получения теоретических свойств результата предполагается, что датчик выдаёт независимые случайные величины, равномерно распределенные на $[0, 1]$.
- Пример: метод обратных функций.
Пусть нужно сгенерировать случайную величину со строго монотонной функцией распределения $F(x) = P(\xi < x)$. Тогда $\xi = F^{-1}(\alpha)$ будет иметь нужное распределение.
Действительно,
$$P(\xi < x) = P(\xi < x) = P(F^{-1}(\alpha) < x) = P(\alpha < F(x)) = F(x),$$
так как по условию α имеет равномерное распределение на $[0, 1]$.
- Равномерное распределение в области можно генерировать методом отбора — описать вокруг фигуры прямоугольник $[a, b] \times [c, d]$, разыгрывать точки как $(a + \alpha_1(b - a), c + \alpha_2(d - c))$, но брать из них только те, которые попадают внутрь фигуры. Понятно, что для генерации α_1 и α_2 получаем с помощью генератора псевдослучайных чисел.

Пусть нам нужно генерировать дискретное распределение, заданное таблицей

$$\xi \sim \begin{pmatrix} v_1 & v_2 & \dots & v_k \\ p_1 & p_2 & \dots & p_k \end{pmatrix}$$

где $\sum_{i=1}^k p_i = 1$

Алгоритм

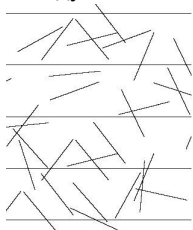
- Считаем накопленные вероятности $f_m = \sum_{i=1}^m p_i$, $m = 1, \dots, k$ ($f_k = 1$). Эти числа разбивают отрезок $[0, 1]$ на k частей, каждая длины p_m .
- Получаем α . Результат (ξ) — номер отрезка, куда попало α .

- Моделировать поведение цен акций и определять, насколько успешна стратегия (покупать-продавать)
- Моделировать движение транспорта в городе, чтобы определять узкие места, пробки и прочее.
- Моделировать поток частиц в газе, чтобы понимать, как распределится температура на поверхности самолета при его полете.
- Моделировать выбросы в атмосфере, предсказывать озоновые дыры, просчитывать последствия от повышения температуры.
- Строить экономические модели, рассчитывать вероятности кризиса, стагнации, роста, ...

Метод Монте-Карло: игла Бюффона

Одно из первых использований метода Монте-Карло (вычисление числовых характеристик с помощью статистического моделирования) является оценка числа π .

Бросается игла длины L на разлинованную бумагу с расстоянием между линиями r , $r \geq L$, случайным образом:



Тогда вероятность того, что игла пересечет линию, равна $\frac{2L}{r\pi}$.

Вычисление интегралов методом Монте-Карло

Есть реальные задачи, когда нужно считать, например, интегралы, а аналитически их невозможно сосчитать.

Пример: есть данные по скорости $v(t)$, нужно сосчитать расстояние, которое машина проехала за время $[0, T]$. Ответ — $\int_0^T v(t)dt$.

Простейший метод Монте-Карло для $I = \int_0^1 v(t)dt$:

- Параметр N — число реализаций.
- Генерируем $\alpha_1, \dots, \alpha_N$ р.р. на $[0, 1]$.
- Оценка интеграла $\hat{I} = \sum_{i=1}^N v(\alpha_i)/N$.

Свойства оценки:

- Несмещенность: $E\hat{I} = I$
- Состоятельность в средне-квадратическом: $D\hat{I} \sim \sigma^2/N$ (если $\int_0^1 v^2(t)dt < \infty$)).
- Асимптотическая нормальность, отсюда можно построить доверительный интервал.

Метод Монте-Карло: уменьшение трудоемкости

Размер асимптотического доверительного интервала для интересующего нас интеграла имеет вид $(\hat{I} \pm c_\gamma \sigma / \sqrt{N})$, где константа в дисперсии σ^2 зависит от метода оценки, $D\hat{I} \sim \sigma^2/N$.

Если хотим получить размер доверительного интервала ε , то $2c_\gamma \sigma / \sqrt{N} = \varepsilon$. Отсюда N берем равным $(2c_\gamma)^2 \sigma^2 / \varepsilon^2$. В наших руках — константа у дисперсии σ^2 , которая определяется алгоритмом.

Трудоемкость равна числу итераций (которое пропорционально дисперсии), умноженной на сложность одной итерации. Если считать, что итерации имеют примерно одинаковую сложность, то получаем задачу уменьшения дисперсии.

Метод Монте-Карло: уменьшение трудоемкости

Класс алгоритмов для оценки $\int_0^T v(t)dt$:

- Параметр N — число реализаций.
- Генерируем $\xi_1, \dots, \xi_N$ с плотностью $p(t)$ на $[0, T]$.
- Оценка интеграла $\hat{I} = \left(\sum_{i=1}^N v(\xi_i) / p(\xi_i) \right) / N$.

Вычисление площади фигуры — частный случай, когда $p(t)$ — равномерное распределение на описанном вокруг фигуры прямоугольнике.

Метод уменьшения дисперсии: выбирать больше точек там, где больше (по модулю) подинтегральная функция (например, с плотностью $p(t) = v(t) / \int_0^T v(s)ds$, если $v(t)$ неотрицательная; но на практике такое не реализуемо, так как нужно уметь считать интеграл, который мы хотим оценить, так как не умеем считать. Поэтому берут что-то близкое, от чего умеют считать интеграл).