

Лекция №3 Статистические методы расчётов

Бросая камешки в воду,
смотри на круги, ими образуемые.
Иначе такое занятие будет пустой
забавою.

(Козьма Прутков)

1. Историческая справка

Первые попытки применения статистических методов расчётов связаны с проверкой основных положений теории вероятностей. Если подбрасывать монету много раз, то вероятность выпадения её одной и той же стороной вверх равна 0,50. Надо отдать должное первым экспериментаторам, проверившим это предсказание теории: французскому естествоиспытателю Бюффону (18 век) и английскому статистику К.Пирсону (19 век). Результаты их опытов представлены в таблице

число испытаний	Бюффон		Пирсон			
	4040	вероят- ность	12000	вероят- ность	24000	Вероят- ность
"орел"	2048	0,507	6019	0,502	12013	0,501
"решетка"	1992	0,493	5981	0,498	11987	0,499

2. Современные проблемы

Такие методы статистических испытаний (или как их теперь называют методы Монте-Карло) получили достаточно широкое распространение именно в настоящее время по ряду причин.

Во-первых, появились ЭВМ, и эта весьма рутинная работа может быть поручена им. Во-вторых, ученые поняли, что не всегда задачи можно решать "строго" в механистическом смысле. Например, для большого числа частиц, взаимодействующих между собой, число уравнений, описывающих динамику такой системы, становится столь большим, что не под силу даже современной ЭВМ. В-третьих, есть задачи, вообще не решаемые традиционными методами. Например, подсчитать площадь облака на фотографии, полученной из космоса.

Традиционные способы подсчёта площади фигуры (вычисление определенного интеграла) здесь не годятся, так как контуры облака невозможно описать какой-либо функцией. Можно конечно, задать эту функцию таблицей значений, однако, опять-таки, контуры облака остаются неявными или размытыми.

В математике появилось новое направление «теория неявных (размытых) множеств». По-английски, «fuzzy set» - пушистое множество. Аналогичные задачи возникают при определении размеров галактик и т.д.

3. Интегрирование методом Монте-Карло

Рассмотрим один из примеров, а именно, вычисление определённого интеграла методом Монте-Карло.

С геометрической точки зрения нужно определить площадь под графиком функции. Будем хаотически ставить точки на плоскости XOY и подсчитывать число попаданий под график функции. Отношение числа попаданий N_s к общему числу бросков N , умноженное на общую площадь рисунка S_0 и будет значением интеграла.

$$S = \int_a^b f(x) dx \approx S_0 \frac{N_s}{N}$$

Рассмотрим достаточно простую функцию $y = \sin(x)$ на интервале $[0; \pi]$. Результат достаточно легко вычисляется аналитически, так что есть, с чем сравнивать.

$$S = \int_0^{\pi} \sin x dx = -\cos x \Big|_0^{\pi} = -\cos(\pi) + \cos(0) = 2,00$$

Алгоритм и текст программы приведены ниже. Программа выполнялась с помощью компиляторов "Power Basic v.2.0", "MS DOS Quick Basic v.1.1", "Visual Basic". Результаты выполнения представлены в таблице.

Число испытаний	PBASIC	QBASIC	VBASIC
50	2.07	1.69	1.69
100	2.13	1.82	2.20
500	2.02	2.12	2.07
1000	2.06	2.02	2.019
10000	2.0091	2.01	2.013

Конечно, при каждом испытании результаты получаются несколько отличными. Разница в выполнении связана с тем, что генераторы случайных чисел в данных компиляторах, являются скорее генераторами «псевдослучайных» чисел.

Конечно, когда речь идет об одномерном интеграле, удобнее использовать обычные известные методы, но когда нужно, например, считать многомерный интеграл, то метод Монте-Карло выглядит предпочтительнее.

Погрешность метода Монте-Карло

$$\sigma \sim \frac{1}{\sqrt{N}}$$

Поэтому рост числа испытаний не приводит к существенному росту точности. Это только один из примеров. На нём основаны задачи случайных блужданий и т.д. (см., например, Гулд, Тобочник)

4. Алгоритм

алгоритм программы выглядит достаточно просто



5. Программа

Программа написана на весьма древнем диалекте BASICa

```
REM "Это программа для вычисления определенного
REM интеграла методом Монте-Карло"
DEF fnf (x) = SIN(x)           'определение функции
INPUT "число испытаний=", N   'число испытаний
'задание границ рассматриваемой области
ax = 0                         'нижняя граница по X
bx = 4 * ATN(1)                'верхняя граница по X
ay = 0                         'нижняя граница по Y
by = 1                         'верхняя граница по Y
```

```

ploshad=(bx - ax) * (by - ay) 'вычисление общей площади
'расчёт и запись значений для построения графика
M = 100 'число точек графика
shagx=(bx - ax)/M 'шаг по x для построения функции
OPEN "monte1.dat" FOR OUTPUT AS #1 'открытие файла для записи
функции
FOR i = 1 TO M 'начало цикла для расчёта функции
xx = shagx * i 'определение X
yy = fnf(xx) 'вычисление функции
PRINT #1, USING "#####.###"; xx; yy 'печатать в файл координат
точки
NEXT i 'конец цикла для расчёта функции
CLOSE 1 'закрытие файла
sum = 0
OPEN "monte2.dat" FOR OUTPUT AS #2 'открытие файла для
построения точек
FOR i = 1 TO N 'начало цикла
x=ax+(bx-ax)*RND 'задание случайного числа X
y=ay+(by-ay)*RND 'задание случайного числа Y
PRINT #2, USING "#####.###"; x; y 'печатать в файл координат
случайной точки
IF y < fnf(x) THEN
sum = sum + 1
ELSEIF y = fnf(x) THEN
sum = sum + 1
ELSE
PRINT "мимо"
END IF
NEXT i 'конец цикла
CLOSE 2 'закрытие файла
integral=ploshad * sum / N 'вычисление интеграла
PRINT "значение интеграла ="; integral
INPUT "нажми на кнопку ENTER", fff
END 'конец программы

```

6. Проверка

Вполне удовлетворительные результаты получаются при проверке условия нормировки распределения Максвелла по приведенным скоростям

$$f(u) = \frac{4}{\sqrt{\pi}} e^{-u^2} u^2, \quad \text{где } u = \frac{v}{v_{\text{вер}}}, \quad v_{\text{вер}} = \sqrt{\frac{2kT}{m}}$$

Напомним, что интеграл (площадь под графиком) в этом случае должна получаться равной единице.

Результаты опытов:

	$N=50$	$N=1000$	$N=1050$
1	0,939	0,965	1,011
2	1,147	0,997	0,992
3	1,095	1,027	1,026
4	1,043	0,996	1,008
5	1,251	0,991	1,043
6	0,834	1,061	1,008
7	1,199	1,017	0,994
8	1,251	0,970	1,011
9	0,938	0,983	0,989
10	1,043	1,069	1,008
ср	1,074	1,0076	1,0048

