Анета Караиванова

СТОХАСТИЧНИ ЧИСЛЕНИ МЕТОДИ И СИМУЛАЦИИ

София, 2012

Стохастични числени методи и симулации
ISBN 978-954-9526-78-3

Издателство Деметра ЕООД, 2012

ПРЕДГОВОР

Предложената на вниманието на читателя книга „Стохастични числени методи и симулации“ е свързана с едноименния курс лекции, четени от автора на студентите от специалност статистика в Софийски университет „Св. Климент Охридски“, Факултет по математика и информатика. Това е и причината за липсата на уводна глава, съдържаща основни понятия и факти от теория на вероятностите и статистиката. Изложеният материал е съобразен с практиката на водещите световни университети и съвременното състояние в областта. Макар и предназначена главно за докторанти и студенти, книгата може да бъде интересна и полезна за всички, които се интересуват от съвременните достижения в областта на стохастичните числени методи и симулации.

Подготовката на книгата и нейното отпечатване са субсидирани от Фонд „Научни изследвания“ в рамките на договорите ДЦВП 2/1 (Развитие на Център за върхови постижения „Суперкомпютърни приложения“) и ДО02-146 („Иновативни квази Монте Карло грид пресмятания – среда, библиотеки, пилотни грид приложения“).

Използвам случая да изразя благодарност на всички колеги, с които съм работила по време на дългогодишната си преподавателска и научно-изследователска дейност за чудесната атмосфера и за оказваното ми от тях съдействие. Специално бих искала да спомена ползотворните дискусии с колегите София Ивановска, Тодор Гюров и Емануил Атанасов (всички от Института по информационни и комуникационни технологии към БАН).

А. Караиванова

СЪДЪРЖАНИЕ

Глава 1. Увод 7

Глава 2. Симулиране на вероятностни разпределения . 11

2.1. Симулиране на равномерно разпределена случайна величина 12

2.1.1. Генератори на псевдослучайни числа. Основни свойства 15

2.1.2. Тестване на генераторите 17

2.1.3. Най-често използваните съвременни генератори на псевдослучайни числа 19

2.2. Симулиране на дискретни случайни величини 22

2.2.1. Моделиране на събития 27

2.2.2. Моделиране на целочислени случайни величини 28

2.3. Симулиране на непрекъснати случайни величини 31

2.3.1. Симулиране на непрекъснати случайни величини по метода на обратната функция 31

2.3.2. Симулиране на непрекъснати случайни величини по метода на селекцията 34

2.3.3. Интерполация на вероятностна плътност на разпределение 38

2.3.4. Оценяване на вероятностна плътност с хистограми 41

Глава 3. Квазислучайни редици **43**

3.1. Квазислучайни генератори 47

3.1.1. Редици, основани на ирационални числа 47

3.1.2. Редица на Холтън 47

3.1.3. Точково множество на Хамерсли 49

3.1.4. (t, m, s)–мрежи и (t, s)–редици 49

3.1.5. Редица на Собол 50

3.1.6. Редица на Фор 52
3.1.7. Точкови множества от тип решетка 53
3.2. Оценки на дискрепанс 54
3.3. Разбъркани редици 55
3.3.1. Разбъркване на редицата на Холтън 56
3.3.2. Разбъркване на редицата на Фор 59
3.3.3. Разбъркване на редицата на Собол 61
Глава 4. Монте Карло методи 63
4.1. Точност на обикновения Монте Карло метод 64
4.2. Сравнение с мрежовите методи 67
4.3. Намаляване на дисперсията 68
4.3.1. Симетризация на подинтегралната функция 69
4.3.2. Отделяне на главната част 69
4.3.3. Метод на съответстващите моменти 70
4.3.4. Стратификация 72
4.3.5. Метод на съществената извадка 75
4.3.6. Метод на разделяне по важност 76
4.3.7. Руска рулетка 78
4.3.8. Адаптивни алгоритми 79
Глава 5. Квази-Монте Карло методи 81
5.1. Неравенство на Коксма-Хлавка 81
5.2. ANOVA декомпозиция и ефективна размерност 85
5.3. Гладкост и намаляване на размерността 87
Глава 6. Паралелни пресмятания 93
Литература 97

Глава 1

Увод

Стохастичните числени методи, известни под името *методи Монте Карло*, са числени методи за решаване на математически задачи с помощта на моделиране на случайни величини и/или случайни функции и статистическа оценка на техните характеристики.

Тези методи са мощен апарат за решаване на много задачи в областта на изчислителната математика, физиката, химията и инженерните науки. Монте Карло методите предлагат простота на конструкциите и често се използват за симулация на процеси, чието поведение може да се интерпретира само в статистически смисъл. Съществуват, обаче, широк клас задачи, за които Монте Карло методите са единствените възможни числени методи за решаване. Независимо от универсалността на Монте Карло методите, техен сериозен недостатък е слабата им сходимост, основана върху грешка с порядък $O(N^{-1/2})$ при статистическа извадка с размер N, когато в оценката на грешката не се използва информация за гладкостта на данните. В резултат от комбинацията от простота на конструкцията, широк обхват на приложимост и бавна сходимост, за Монте Карло пресмятанията се използва твърде много компютърно време. Например, според данни от мониторирането на Европейската грид инициатива (www.egi.eu) около 80 процента от общото процесорно време се използва от грид приложенията с Монте Карло пресмятания. Според данни от Департамента по енергетика на САЩ около 70 процента от компютърното време на компютрите в департамента се използва за Монте Карло пресмятания.

Това е едно предизвикателство за изследователите в областта на Монте Карло методите. Дори скромни подобрения в тези методи имат съществен принос по отношение на ефективността и обхвата им на приложимост.

Всъщност, голяма част от усилията в разработването на Монте Карло методи са насочени към конструиране на методи с намалена дисперсия, които водят до ускоряване на сходимостта чрез намаляване на константата пред $O(N^{-1/2})$ в оценката за грешката. Един друг подход за подобряване на сходимостта на Монте Карло методите е да се промени вида на използваната редица. *Квази-Монте Карло методите* използват квазислучайни редици (известни също под името редици с малък дискрепанс) вместо обичайните псевдослучайни редици. Последвалото изследване и развитие на тази група методи доведе до разработването на *рандомизирани квази-Монте Карло методи*, които използват рандомизирани (разбъркани) редици с малък дискрепанс.

Един от основните способи за построяване на методи Монте Карло е свеждането на задачата към някакво интегрално представяне, което се интерпретира като математическо очакване (включително условно) на случаен елемент (случайна величина, случайна функция). Този елемент се нарича оценка на търсената величина или функция. За пресмятане на необходимата стойност се използва закона на големите числа, от който следва, че за приближена стойност на математическото очакване на случайната величина ξ се използва средното аритметично

$$E\xi \approx \frac{\xi_1 + \ldots + \xi_n}{n}$$

за достатъчно голямо n; тук $\xi_i, i = 1, \ldots, n$ са независими реализации на случайната величина ξ.

Въвеждането на случаен елемент ξ за числено приближение на търсената величина с метод Монте Карло се нарича рандомизация на задачата. Трябва да отбележим, че понятието оценка в теорията на методите Монте Карло се отличава малко от аналогичния термин в математическата статистика. Така, в горната формула случайната величина ξ е оценката на търсената величина в термините на методите Монте Карло, а средното

$$\overline{\xi_n} = \frac{\xi_1 + \ldots + \xi_n}{n}$$

е оценката на математическото очакване $E\xi$ в термините на математическата статистика. Изборът на оценката ξ, като правило, е нееднозначен, поради което основните въпроси при използването на методите за числено статистическо моделиране, са: избор на оптимална оценка ξ и разработка на алгоритми за намиране на стойностите от извадката $\{\xi_i\}$ с помощта на компютър.

Може би най-ранното документирано използване на случайна извадка за намиране на приближена стойност на интеграл, е това на Конт де Буфон (1707–1788). През 1777 той описва следния експеримент. Игла с дължина L се хвърля случайно върху хоризонтална равнина с разграфени прави линии на разстояние $d, (d > L)$. Каква е вероятността P иглата да пресече една от тези линии? За да определи P, Конт де Буфон извършва многократно експеримента хвърляне на игла. Математическият анализ, който той извършва на задачата, показва, че

$$P = \frac{2L}{\pi d}.$$

Няколко години по-късно, Лаплас предлага горната тази формула да се използва за пресмятане на π, като се използва експерименталната оценка за вероятността. Това е наистина метод Монте Карло за определяне на π; скоростта на сходимост, обаче, е бавна. Експериментът може да бъде направен онлайн на адрес:

`http://webspace.ship.edu/deensley/mathdl/stats/Buffon.html`

Много открития бяха направени в теория на вероятностите и теорията на случайните процеси, които са използвани в теоретичната основа на Монте Карло. Например, Курант, Фридрихс и Леви доказват еквивалентността на поведението на определени случайни процеси и решението на частни диференциални уравнения. През 1930 Енрико Ферми прави числени експерименти, които по-късно се наричат Монте Карло пресмятания, в изследванията си на новооткрития неутрон.

През Втората световна война, съвместната работа на големи учени като Фон Нойман, Ферми, Улам и Метрополис и появата на съвременните дигитални компютри даде силен тласък на развитието на Монте Карло теорията. Даже терминът Монте Карло е измислен от учените, работещи по разработването на ядрени оръжия в Лос Аламос през 1940-те. В края на 40-те и началото на 50-те години на миналия век има взрив на интерес; появяват се много статии, описващи новия метод и как той може да бъде използван за решаване на задачи в областта на статистическата механика, радиационния транспорт, икономическите модели, и др. За съжаление, по това време компютрите не могат да извършват нещо повече освен пилотни пресмятания в много области. По-късното развитие на компютърната техника позволява извършването на все по-интензивни пресмятания. Последвалите успехи отговарят на оптимистичните очаквания на пионерите от средата на 20-ти век.

Глава 2

Симулиране на случайни величини

Компютърното симулиране на случайни величини и случайни процеси е една най-бързо развиващите се области на изчислителната статистика. Много статистически процедури, методи и алгоритми се основават на симулация на случайни величини. Една традиционна област е използването на случайни числа за получаване на извадка от популация. По-новите приложения включват симулиране на многомерни сложни стохастични системи, които не могат да бъдат анализирани аналитично. В много реални ситуации вероятностните разпределения са твърде сложни и изследователите анализират извадки от тяхната компютърна симулация. При наличие на все по-бърза изчислителна и визуализационна среда напоследък, все повече разпределения стават достъпни посредством симулация.

Тази глава е посветена на методите за симулация на известни разпределения. Генерирането на случайна величина има две стъпки:

1. Генериране на извадки от стандартно равномерно разпределение, например в $(0, 1]$

2. Трансформиране на извадките по подходящ начин за симулиране на желаното разпределение.

.

2.1 Симулиране на равномерно разпределена случайна величина

Да припомним основните характеристики на равномерно разпределена случайна величина α в интервала $[0,1]$: за $0 \leq u \leq 1$ плътността и е $p_\alpha \equiv 1$, функцията на разпределение е $F_\alpha = u$, математическото очакване е $E\alpha = 1/2$ и дисперията е $V\alpha = 1/12$.

Има три основни начина за компютърно симулиране на равномерно разпределена случайна величина: таблици със случайни цифри, датчици на случайни числа (физически датчици) и генератори на псевдослучайни числа. Изборът на един или друг начин се основава на следните критерии:

- статистическите свойства на получаваната извадка; тези свойства се изучават чрез проверка с определени статистически тестове;
- бързината, с която се получава извадката с компютър, и икономичното използване на ресурсите на компютъра.

Съществуват два принципа за построяване на случайни числа: отделно да се моделират случайни цифри (обикновено двоични) и от тях да се съставя стандартно случайно число (по този начин получаваме α с помощта на таблици и датчици на случайни числа), или да получим цялото случайно число (както при генераторите на псевдослучайни числа). Следващото твърдение показва, че тези два начина са равносилни. Резултатът е получен за десетични случайни цифри ε_i, които с еднаква вероятност $1/10$ приемат цели стойности от 0 до 9 (доказателството за двоични случайни цифри е аналогично).

Твърдение 2.1.1 *Десетичните цифри $\varepsilon_1, \ldots, \varepsilon_k, \ldots$ на случайното число*

$$\alpha = 0, \varepsilon_1 \ldots \varepsilon_k \ldots \tag{2.1.1}$$

са независими случайни цифри. Обратно, ако $\varepsilon_1, \ldots \varepsilon_k, \ldots$ са независими случайни цифри, то α в (2.1.1) е стандартно случайно число.

Доказателство. Ако случайната величина (2.1.1) е равномерно разпределена в $[0,1)$, то $\varepsilon_k = i$ тогава и само тогава, когато

$$0, \varepsilon_1 \ldots \varepsilon_{k-1} i \leq \alpha \leq 0, \varepsilon_1 \ldots \varepsilon_{k-1} i + 0, \underbrace{0 \ldots 01}_{k} \tag{2.1.2}$$

при което $\varepsilon_1, \ldots, \varepsilon_{k-1}$ в (2.1.2) могат да приемат произволни цели стойности от 0 до 9. Дължината на интервала (2.1.2) е равна на 10^{-k}, и интервалите (2.1.2) за различни набори $\varepsilon_1, \ldots, \varepsilon_{k-1}$ не се пресичат, поради което

$$P(\varepsilon_k = i) = \sum_{\varepsilon_1,\ldots,\varepsilon_{k-1}=0}^{9} 10^{-k} = 10^{k-1}10^{-k} = 0,1. \qquad (2.1.3)$$

Сега ще докажем независимостта на ε_s и ε_k, където $1 \leq s < k$. За тази цел разглеждаме вероятността $P(\varepsilon_k = i, \varepsilon_s = j)$. Очевидно, тази вероятност може да се разглежда като условната вероятност, че е изпълнено (2.1.2) при условие, че ε_s е фиксирано. Тогава, по аналогия с (2.1.3) имаме

$$P(\varepsilon_k = i, \varepsilon_s = j) = \sum_{\varepsilon_1,\ldots,\varepsilon_{s-1},\varepsilon_{s+1},\ldots,\varepsilon_{k-1}=0}^{9} 10^{-k} - 10^{k-2}10^{k} =$$

$$= 0,01 = P(\varepsilon_k = i)P(\varepsilon_s = j).$$

Аналогично

$$P(\varepsilon_{k_1} = i_1, \ldots, \varepsilon_{k_q} = i_q) = 10^{-q} = P(\varepsilon_{k_1} = i_1) \times \ldots \times P(\varepsilon_{k_q} = i_q),$$

което означава независимост на случайните цифри на числото (2.1.1).

Ще докажем обратното твърдение. Сега предполагаме, че случайните цифри на числото (2.1.1) са независими и ще покажем, че $P(\alpha < x) = x$ за всяко

$$x = 0, a_1 a_2 \ldots a_k \ldots$$

от полуинтервала $[0, 1)$. Ако $\alpha < x$, то в разложението (2.1.1) или $\varepsilon_1 = a_1$ и $\varepsilon_2 < a_2$, или $\varepsilon_1 = a_1, \varepsilon_2 = a_2$ и $\varepsilon_3 < a_3$, и т.н. По този начин

$$P(\alpha < x) = \sum_{k=1}^{\infty} P(\varepsilon_1 = a_1, \ldots \varepsilon_{k-1} = a_{k-1}, \varepsilon_k < a_k) =$$

$$= \sum_{k=1}^{\infty} P(\varepsilon_1 = a_1) \times \ldots \times P(\varepsilon_{k-1} = a_{k-1}) \times P(\varepsilon_k < a_k)$$

(поради незаависимостта на случайните цифри $\varepsilon_1, \ldots, \varepsilon_k$). Лесно е да се види, че $P(\alpha < x) = a_k 10^{-1}$. Тогава

$$P(\alpha < x) = \sum_{k=1}^{\infty} a_k 10^{-1} 10^{-(k-1)} = \sum_{k=1}^{\infty} a_k 10^{-k} = x,$$

и твърдението е доказано.

Да разгледаме предимствата и недостатъците на трите начина за реализация на случайно число:

- **Таблици:** използват се големи масиви предварително получени случайни цифри, от които по определен алгоритъм се избират цифри за запис на поредното случайно число. При съвременните компютърни пресмятания таблиците не се използват по следните причини. Първо, чисто технически изготвянето на такава таблица е сложна задача. Второ, има ограничение за броя на получените случайни числа и възникват периодически повторения на случайните числа (макар и с голям период). Освен това, таблицата заема значително място в паметта на изчислителната система и получаването на случайните числа е твърде бавно. Към предимствата на таблиците се отнася еднократната статистическа проверка на качествата им, както и възпроизводимостта на числата.

- **Датчици:** използват се технически устройства (например, радиоелектронни прибори с шум), които изработват случайна последователност от двоични цифри. Тяхното предимство е неограниченият брой случайни числа, които могат да произведат. Недостатъците (освен скоростта в някои случаи) включват необходимостта периодически да се прави статистическа проверка на произведените числа (даже свръхнадеждно устройство може да има отклонения в работата си). Освен това, няма възпроизводимост. Трябва да се отбележи, обаче, че има изследователи, които предпочитат получени по този начин случайни числа, и работата по конструиране на такива устройства продължава. Например, устройството Quantum Random Bit Generator [60] е разработено в последните години и случайността се основава на вътрешната случайност на квантов физически процес на фотонна емисия в полупроводници. За дистанционното му използване е разработена уеб услуга Quantum Random Bit Generator Service (QRBGS).

- **Генератори на псевдослучайни числа:** Монте Карло пресмятанията използват най-често числата, произведени от генератори на

псевдослучайни числа, които представляват компютърни програми. Аргументите в полза на тези генератори са възможността да се възпроизведат пресмятанията, бързината, с която се получават числата, отсъствието на външни устройства, липса на необходимост за многократна проверка на качеството на числата, незначително натоварване на паметта. Обаче, и тук изборът на псевдослучайните числа е ограничен, и се проявяват ефектите, свързани с периодичността.

2.1.1 Генератори на псевдослучайни числа. Основни свойства

Разглеждаме следния общ вид на генератор на псевдослучайни числа (ГПСЧ):

$$\alpha_{n+1} = \Phi(\alpha_n), \tag{2.1.4}$$

където началната стойност α_0 е зададена. Областта на стойностите на $\Phi(.)$ е полуинтервала $[0, 1)$. Освен това, от съображения, че, от една страна, точките

$$(\alpha_1, \alpha_2 = \Phi(\alpha_1)), (\alpha_3, \alpha_4 = \Phi(\alpha_3)), (\alpha_5, \alpha_6), \ldots$$

лежат върху кривата $y = \Phi(x)$, а от друга – тези точки трябва да бъдат равномерно разпределени в квадрата $\{(x, y) : 0 \leq x < 1, 0 \leq y < 1\}$, следва че графиката на тази функция трябва плътно да запълва този квадрат. Един пример за такава функция $\Phi(.)$ е функцията

$$\Phi_1(x) = \{Mx\} \tag{2.1.5}$$

за голям множител M, където $\{A\}$ означава дробната част на числото A.

Алгоритъм на фон Нойман. Друг пример на $\Phi(.)$ е:

$$\Phi_2(x) = 10^{-2k}[10^{2k}\{10^k x^2\}], \tag{2.1.6}$$

където $[A]$ означава цялата част на числото A. Тази функция служи за моделиране на $2k$-значни случайни числа по *метода на средата на квадрата на Джон фон Нойман*, чиято интерпретация е следната: ако

$$\alpha_n = 0, a_1 a_2 \ldots a_{2k},$$

то

$$\alpha_{n+1} = 0, b_{k+1} b_{k+2} \ldots b_{3k},$$

където b_i са съответните цифри от числото

$$\alpha_n^2 = 0, b_1 b_2 \dots b_{4k}.$$

Това е първият генератор на псевдослучайни числа, предложен от фон Нойман в средата на миналия век. По-късно, при изследването му се оказва, че той произвежда голям брой прекалено малки числа.

Твърдение 2.1.2 *Случайната величина $\beta = \{M\alpha\}$ е равномерно разпределена върху полуинтервала $[0, 1)$ за всяко цяло положително число M.*

Доказателство. Ако $x \in [0, 1)$, то вероятността

$$P(\beta < \alpha) = \sum_{k=0}^{M-1} P(k \leq M\alpha < k + x) =$$

$$\sum_{k=0}^{M-1} P(kM^{-1} \leq \alpha < (k + x)M^{-1}) = \sum_{k=0}^{M-1} xM^{-1} = x,$$

което и трябваше да се докаже.

От последното твърдение следва, че разглежданият метод наистина произвежда редица от случайни числа. Очевидно е, обаче, че съседните членове на редицата са взаимнозависими. Възниква въпросът: колко голяма е тази зависимост?

Твърдение 2.1.3 *Коефициентът на корелация на случайните величини α и $\beta = \{M\alpha\}$ е равен на $1/M$ за произволно положително M.*

Доказателство. От предишното твърдение следва, че случайната величина β е също стандартна случайна величина и коефициентът на корелация е

$$\rho(\alpha, \beta) = E\left(\left(\frac{\alpha - 1/2}{\sqrt{1/12}}\right)\left(\frac{\beta - 1/2}{\sqrt{1/12}}\right)\right) = E\left(\left(\frac{M\alpha - M/2}{M\sqrt{1/12}}\right)\left(\frac{\beta - 1/2}{\sqrt{1/12}}\right)\right) =$$

$$E\left(\left(\frac{[M\alpha] + \{M\alpha\} - (M/2 - 1/2) - 1/2}{M\sqrt{1/12}}\right)\left(\frac{\beta - 1/2}{\sqrt{1/12}}\right)\right) =$$

$$E\left(\left(\frac{\gamma-(M/2-1/2)}{M\sqrt{1/12}}\right)\left(\frac{\beta-1/2}{\sqrt{1/12}}\right)\right)+E\left(\left(\frac{\beta-1/2}{M\sqrt{1/12}}\right)\left(\frac{\beta-1/2}{\sqrt{1/12}}\right)\right),$$

където $\gamma=[M\alpha]$. Случайните величини γ и β са независими (следва от първото твърдение), и от което следва, че

$$\rho(\alpha,\beta)=\frac{\sqrt{V\gamma}}{M\sqrt{1/12}}\rho(\gamma,\beta)+\frac{1}{M}\rho(\beta,\beta)=\frac{1}{M},$$

което и трябваше да се докаже.

От последното твърдение следва, че за голямо M, коефициентът на корелация между съседни членове на редицата, не е голям.

2.1.2 Тестване на генераторите

Веднага ще отбележим, че никаква система тестове не е достатъчна, за да се обяви един или друг генератор за безупречен. Процесът на проверка на конкретен генератор е безкраен. Нещо повече, всяка задача, при численото решаване на която се използва генератор на случайни (псевдослучайни) числа, може да се разглежда като тест за този генератор. Някои от най-разпространените тестове са следните:

Най-често употребяван е критерия χ^2, чиято същност се състои в следното: Нека в резултат на провеждането на N независими експеримента (в нашия случай, един експеримент е един такт на генератора) сме получили стойности $\xi_1,\ldots,\xi_N$ и трябва да се изясни дали те са извадка от стойностите на случайната величина ξ със зададен закон за разпределение и област на стойности (в нашия случай $\xi=\alpha$, където α е стандартно случайно число, и $X=[0,1)$). Разделяме множеството X на τ непресичащи се по двойки множества $X=X_1+\ldots+X_\tau$ (в нашия случай може да разделим $X=[0,1)$ на полуинтервали $X_i=\left[\frac{i-1}{r};\frac{i}{r}\right)$) и по зададения закон на разпределение пресмятаме вероятностите $p_i=P(\xi\in X_i)$ (в нашия случай, $p_i=1/r$)). След което се прави групиране на стойностите: за всяко $i, i=1,\ldots,r$ пресмятане броя ν_i на стойностите ξ_i, попадащи в X_i.

Ако $\{\xi_i\}$ наистина са стойности на извадка на ξ, то $E\nu_i=Np_i$ и е в сила следното твърдение:

Твърдение 2.1.4 *(Теорема на К. Пирсон). Каквито и да се величината ξ и разбивката $X=X_1\cup\ldots\cup X_\tau$ (такава, че всички $p_i>0$), за всяко $x>0$*

$$\lim_{N\to\infty} \Pr(\chi_N^2 < x) = \int_0^x k_{r-1}(u)du,$$

където

$$, \chi_N^2 = \sum_{i=1}^{r} \frac{(\nu_i - Np_i)^2}{Np_i},$$

$k_m(.)$ *е плътността на разпределението* χ^2 *с* m *степени на свобода*

$$k_m(u) = \frac{u^{m/2-1}e^{-u/2}}{2^{m/2}\Gamma(m/2)},$$

а $\Gamma(.)$ *е гама-функцията*

$$\Gamma(v) = \int_0^\infty w^{v-1}e^{-w}dw, \;\; v > 0.$$

Последното твърдение дава следната процедура за проверка на стойностите $\xi_1, \ldots, \xi_N$ за достатъчно голямо N. Задаваме доверителна вероятност (коефициент на доверие) β (обикновено $\beta = 0.95, 0.99, 0.999$) и от уравнението

$$\int_{\chi^2(r-1)(1-\beta)}^{\infty} k_{r-1}(u)du = 1 - \beta$$

определяме (обичайно с помощта на съответните таблици) величината $\chi^2(r-1)(1-\beta)$, която се нарича доверителна граница с ниво на значимост $(1-\beta)$. Отчитайки получените стойности на p_i и ν_i, пресмятаме χ_N^2 по формулата от теоремата на Пирсон, и ако

$$\chi_N^2 < \chi^2(r-1, 1-\beta),$$

то изследваната извадка $\{\xi_j\}$ се приема за удовлетворителна, а ако

$$\chi_N^2 \geq \chi^2(r-1, 1-\beta),$$

за неудовлетворителна. Ако $\beta = 0.95$, то стойността на χ_N^2 се нарича почти значима, при $\beta = 0.99$ – значима. а при $\beta = 0.999$ – силно значима.

Тъй като Монте Карло методите са много ефективни за оценка на интеграли, важно значение има проверката на свойството *k-равномерност*, смисълът на която се състои в това, че векторите

$$(a_1, \ldots, a_k), (a_{k+1}, \ldots, a_{2k}), \ldots, (a_{k(N-1)+1}, \ldots, a_{Nk})$$

трябва с нарастването на N с вероятност 1 да запълват единичния k-мерен куб. Проверката на това свойство може също да се осъществи с помощта на критерия χ^2: в този случай областта от стойности X е единичния k-мерен куб, който може да се разбие на $r = s^k$ еднакви кубчета (като се въведе равномерна мрежа със стъпка $1/(s-1)$ по всяка координата), пресметне се честотата на попадане на векторите от горния вид в тези малки кубчета и съответната стойност на χ^2_N по формулата от теоремата на Пирсон, и т.н.

За проверка на качествата на случайните числа се използват и други критерии – например, критерий на Смирнов, корелационни критерии, и т.н.

2.1.3 Най-често използваните съвременни генератори на псевдослучайни числа

Основните видове генератори на псевдослучайни числа, които се използват в съвременните пресмятания са: линейни конгруентни генератори, генератори с изместени регистри, адитивни и мултипликативни генератори на Фибоначи, комбинирани генератори и конгруентни генератори, използващи обратна функция.

- **Конгруентни генератори**

  Конгруентните генератори използват модулна аритметика. Да припомним означенията в модулната аритметика. За положително цяло число m и дадено число b, a се нарича остатък на b по модул m, ако

  $$a = b - \lfloor b/m \rfloor m,$$

  където $\lfloor . \rfloor$ означава най-голямото цяло число, по-малко от аргумента. Например, ако $m = 5$ и $b = 13$, тогава остатъкът на b по модул m е 3. Ще използваме означението, че a е $b \mod m$. Може да се провери, че голям брой числа имат същия остатък. В действителност, ако дефинираме, че две числа са еквивалентни, ако имат един и същ остатък, то тази релация създава един клас на еквивалентност върху множеството на числата. Ограничаваме се с неотрицателните цели числа и в този случай класът на еквивалентност за това множество е

  $$\{0, 1, 2, 3, \ldots, m-1\}.$$

Модулната аритметика се използва за генериране на псевдослучайни числа по следния начин. Дефинираме следващия елемент от редицата като

$$x_{i+1} = (ax_i + c) \pmod{m}$$

където a се нарича множител, m - основа, и c – адитивна константа. Генераторите, основани на тази формула, се наричат *линейни конгруентни генератори.*

Ако $c = 0$, генераторът се нарича мултипликативен генератор. Изборът на a и m определя свойствата на редицата. Започваме с начална стойност, $x_0 \in 1, 2, \ldots, m-1$, наричана ядро на генератора. Следващите елементи на редицата се определят от релацията:

$$x_{i+1} = (ax_i) \pmod{m}$$

Трябва да се отбележи, че тази релация е напълно детерминистична, т.е., ако едно число се повтаря в редицата, то следващата след него редица също се повтаря. По-нататък, тъй като елементите на редицата лежат в множеството $x_0 \in \{1, 2, \ldots, m-1\}$, числата се повтарят, ако генерираме достатъчно дълга редица. Това води до дефиницията на период.

Дефиниция 2.1.5 *Периодът на редица се дефинира като броя на елементите от редицата, преди редицата да повтори себе си.*

За основа m максималният период на редицата е $m - 1$. За едно и също m, но за различни множители a, резултантната редица може да има съвсем различни периоди.

- **Генератори с изместени регистри**

  Този клас от генератори се основава на бързо изпълнение от устройство с крайна памет, което използва претеглена сума от последните n елемента на реда за пресмятане на следващия елемент. Общата форма може да бъде описана чрез уравнението:

$$x_{i+1} = w_1 x_i + w_2 x_{i-1} + \ldots + w_n x_{i-n+1}.$$

  Предимството на този метод е, е може да се изпълнява хардуерно сравнително бързо.

- **Генератори на Фибоначи** (Lagged Fibonacci Generators)

  Общата форма на тези генератори е:

$$x_i = x_{i-r}.x_{i-s},$$

  където . е двоична операция (събиране или умножение), дефинирана върху множеството, от което x_i взема стойности.

  Формулата, задаваща адитивният генератор на Фибоначи, е:

$$z_n = z_{n-s} + z_{n-r} \pmod{2^k};\ r > s,$$

  а мултипликативният генератор се дефинира от:

$$x_n = x_{n-s} \times x_{n-r} \pmod{2^k};\ r > s.$$

Изследванията върху генераторите са довели до следните изводи за тяхната изчислителна сложност и качество:

1. Рекурсиите по модул две-на-степен са изчислително евтини, но имат твърде проста структура, която влошава качеството на произвежданите числа.

2. Рекурсиите по модул просто число са по-скъпи изчислително, но имат по-високо качество. Напоследък се използват прости числа на Мерсен от вида $2^p - 1$, където p е просто число.

3. Генераторите с измесстени регистри (например, Mersenne Twisters) са изчислително ефективни и имат добро качество.

4. Генераторите на Фибоначи са изчислително ефективни, но имат структурни недостатъци.

5. За комбинираните генератори (обикновено генератор с изместени регистри се комбинира с генератор на Фибоначи) може да се докаже, че са добри.

6. Модулните инверсии са изчислително скъпи.

7. Максималните периоди на основните генератори са:

   Линеен конгруентен ($x_n = ax_{n-1} + c \pmod{m}$) : Ако m е просто число, то максималният период на редицата е $\mathrm{Per}(x_n) = m - 1$. Ако $m = 2^k$, то $\mathrm{Per}(x_n) = 2^k$ при $c \neq 0$, и $\mathrm{Per}(x_n) = 2^{k-2}$ при $c = 0$.

Периодът на генератор с изместени регистри ($y_n = y_{n-s} + y_{n-r} \pmod 2$; $r > s$) е $\mathrm{Per}(y_n) = 2^r - 1$.

Периодът на адитивния генератор на Фибоначи ($z_n = z_{n-s} + z_{n-r} \pmod{2^k}; r > s$) е $\mathrm{Per}(z_n) = (2^r - 1)(2^{k-1})$.

Периодът на мултипликативния генератор на Фибоначи ($x_n = x_{n-s} \times x_{n-r} \pmod{2^k}$, $r > s$) е $\mathrm{Per}(x_n) = (2^r - 1)(2^{k-3})$.

Периодът на комбинирания генератор ($w_n = y_n + z_n \pmod p$) е $\mathrm{Per}(w_n) = \mathrm{lcm}(\mathrm{Per}(y_n), \mathrm{Per}(Z_n))$, където lcm означава най-малкия общ множител.

В горните формули членовете на редиците, формирани от различните генератори, са означени с различни букви, за да е ясно кои генератори формират комбинирания генератор.

2.2 Симулиране на дискретни случайни величини

Една дискретна случайна величина приема само изброимо много стойности с предефинирани вероятности. Една дискретна случайна величина се характеризира чрез функция на вероятностно разпределение, дефинирана като

$$\Pr(\xi = x_1) = p_1$$

$$\Pr(\xi = x_2) = p_2$$

$$\dots$$

$$\Pr(\xi = x_n) = p_n$$

$$\dots$$

така че за всяко i имаме $0 < p_i < 1$ и $\sum_{i=1} p_i = 1$. Най-често използваните дискретни случайни величини са биномна, поасонова, геометрична и отрицателно-биномна. Например, функцията на разпределение на поасонова случайна величина с параметър λ, е

$$p_i = \frac{e^{-\lambda}\lambda^i}{i!}, \quad i = 0, 1, 2, \dots .$$

За биномна случайна величина с параметри (n, p), функцията на вероятностно разпределение е

$$p_i = \binom{n}{i} p^i (1-p)^{n-i}.$$

Преди да разгледаме различни алгоритми за симулиране на дискретни случайни величини, ще дефинираме понятието симулация.

Дефиниция 2.2.1 *За дадена случайна величина с определена вероятностна функция на разпределение*

$$\{(x_i, p_i), i = 0, 1, 2, \ldots\},$$

процесът на избор на стойност x_i с вероятност p_i се нарича симулация.

Ако този избор е направен многократно, генерира се редица $\{X_j\}$, за която:

$$\frac{1}{n} \sum_{j=1}^{n} I_{X_j}(\{x_i\}) \to p_i, \tag{2.2.1}$$

където I_{X_j} е индикаторната функция за редицата. С други думи, ние правим избор от множеството с допустимите стойности с асоциираната вероятност. Съществуват различни стандартни техники за симулация на дискретни случайни величини.

Отначало ще разгледаме стандартния алгоритъм за симулация на случайната величина ξ, същността на който се състои в избор на номер m и стойност x_m на основание на съотношението

$$\Pr\left(\sum_{i=0}^{m-1} p_i \leq \alpha < \sum_{i=0}^{m} p_i\right) = p_m,$$

където α е стандартно случайно число, т.е., реализация на равномерно разпределена случайна величина в интервала $[0, 1]$.

Алгоритъм 1. Симулираме $Q := \alpha$ и полагаме $m := 0$. Присвояваме

$$Q := Q - p_m$$

Ако новото Q не е положително, то за m избираме текущата стойност, в противен случай присвояваме $m := m + 1$ и $Q := Q - p_m$ и отново правим проверка за положителност на Q.

Лесно се вижда, че в случая, когато $\xi = x_m$, се правят $m+1$ проверки за положителност на Q, и само при $\xi = x_n$ се правят n проверки. В общия брой аритметични операции δ на горния алгоритъм (δ е случайна величина) влизат операциите за моделиране на една стандартна случайна величина (ще ги означим с a) и операциите по сравненията на Q с нула (ще ги означим с b_1). Тогава средната трудоемкост на алгоритъма е

$$S = \mathrm{E}[\delta] = a + \left(\sum_{i=0}^{n-1} (i+1)p_i + np_n \right) b_1. \tag{2.2.2}$$

Не е трудно да се покаже, че оптималното разпределение на вероятностите, при които трудоемкостта S е минимална, е

$$p_0 \geq p_1 \geq \ldots \geq p_n,$$

поради което при моделиране на дискретна случайна величина с краен брой стойности е целесъобразно да се разполагат вероятностите p_i по реда на тяхното намаляване.

Симулирането на случайна величина ξ с краен брой стойности значително се опростява, когато всички стойности $x_0, x_1, \ldots, x_n$ са равновероятни, т.е. всички p_i са равни на $\frac{1}{n+1}$ (такова разпределение се нарича *дискретно равномерно*). В този случай за избора на m има проста формула

$$m = \lfloor a(n+1) \rfloor,$$

където $\lfloor A \rfloor$ (както винаги) е цялата част на A.

Алгоритъм 2. Избор „без връщане“. Задаваме масив **A** с размерност N и полагаме

$$\mathbf{A}[k] = k; \ k = 1, \ldots, N.$$

Избираме първия номер m_1. За тази цел по формулата за m от дискретното равномерно разпределение симулираме случаен номер от N номера:

$$q_1 = \lfloor \alpha_1 N \rfloor + 1. \tag{2.2.3}$$

Полагаме $m_1 = \mathbf{A}[q_1] = q_1$. Извършваме присвояването $\mathbf{A}[q_1] := \mathbf{A}[N]$.

Избираме втори номер m_2. За тази цел отначало генерираме случаен номер измежду $(N-1)$ номера:

$$q_2 = \lfloor \alpha_2 (N-1) \rfloor + 1 \tag{2.2.4}$$

и полагаме $m_2 = \mathbf{A}[q_2]$; присвояваме $\mathbf{A}[q_2] := \mathbf{A}[N-1]$.

...

Избираме i-ти номер m_i. За целта, първо генерираме случаен номер измежду $(N-i+1)$ номера:

$$q_i = \lfloor \alpha_i (N-i+1) \rfloor + 1 \tag{2.2.5}$$

и полагаме $m_i = \mathbf{A}[q_i]$; извършваме присвояването $\mathbf{A}[q_i] := \mathbf{A}[N-i+1]$.

...

Избираме n-ти номер m_n, За целта генерираме случаен номер измежду $(N-n+1)$ номера по формулата (2.2.5) за $i = n$ и полагаме $m_n = \mathbf{A}[q_n]$.

Трудоемкостта на Алгоритъм 2 е пропорционална на величината

$$S_1 = n(a + b_2),$$

където a е средното време за компютърното генериране на една стандартна случайна величина α_i, b_2 е средното време за пресмятане на i-тия номер (умножение на α_i с $(N-i)$, вземане на цялата част от резултата и прибавяне на единица) и присвояване.

За случая $n \ll N$ алгоритъм 2 има следната модификация:

Алгоритъм 3. Модифициран метод „без връщане“. Задаваме масив $\mathbf{B}$ с размерност $(n-1)$. Полагаме $\mathbf{B}[k] := 0,\ k = 1, \ldots, n-1$. По формулата (2.2.3) избираме случаен номер q_1 измежду N номера и го обявяваме за първия номер: $m_1 = q_1$. Полагаме $\mathbf{B}[1] := q_1$.

По формулата (2.2.4) избираме случаен номер q_2 измежду $(N-1)$ номера. Ако $q_2 = \mathbf{B}[1]$, полагаме $\mathbf{B}[1] := 0$ и обявяваме $m_2 = N$; в противен случай $m_2 = q_2$. Полагаме $\mathbf{B}[2] = q_2$.

...

Избираме случаен номер q_i измежду $(N-i+1)$ номера по формулата (2.2.5). Проверяваме последователно равенствата:

$$q_i = \mathbf{B}[i-1], q_i = \mathbf{B}[i-2], \ldots, q_i = \mathbf{B}[1]. \tag{2.2.6}$$

Нека кое да е от равенствата (2.2.6) да е изпълнено, т.е., намерен е номер j такъв, че $q_i = \mathbf{B}[i-j]$. Тогава полагаме $\mathbf{B}[i-j] := 0$ и обявяваме i-тия

избран номер $m_i = N - i + j + 1$. Ако нито едно от равенствата (2.2.6) не е изпълнено, то $m_i = q_i$. Полагаме $\mathbf{B}[i] := q_i$.

...

Избираме случаен номер q_s от $(N - n + 1)$ номера по формулата (2.2.5) за $i = n$. Проверяваме последователно равенствата (2.2.6) за $i = n$. Ако намерим $j : q_n = \mathbf{B}[n - j]$ то $m_n = N - n + j + 1$; ако няма такова j, то $m_n = q_n$.

Ако Δ е случайна величина, равна на времето, което е необходимо за проверка на равенствата (2.2.6), то трудоемкостта на Алгоритъм 3 се определя от:

$$\mathrm{E}[\Delta] = (\sum_{i=2}^{n}((i-1)(1-p_i)^{i-1} + \sum_{j=1}^{i-1} jp_i(1-p_i)^{j-1}))c_2, \quad p_i = \frac{i-1}{N-i+1},$$

където c_2 са средните разходи за проверка на едно равенство от типа $q_i = \mathbf{B}[i - j]$.

Алгоритъм 4. Избор с „връщане“. Тук, както и в Алгоритъм 3, декларираме масив $\mathbf{B}$ с размерност $(n - 1)$. За избор на печеливш номер m_1 се генерира случаен номер q_i от N номера по формулата

$$q_i = \lfloor \alpha_i N \rfloor + 1. \tag{2.2.7}$$

По-нататък се проверяват равенствата (2.2.6), при което елементите на масива $\mathbf{B}[1], \ldots, \mathbf{B}[i - 1]$ съдържат печелившите номера $m_1, \ldots, m_{i-1}$, получени на предишните стъпки, по-специално, $\mathbf{B}[1] := m_1 = q_1$. Ако някое от равенствата (2.2.6) е изпълнено, то номерът q_i се отхвърля и се моделира нова стойност q_i по формулата (2.2.7) дотогава, докато не се получи номер, различен от всики предишни (т.е., проверката на всички равенства (2.2.6) дава отрицателен резултат), и тогава $m_i = q_i$ и $\mathbf{B}[i] = m_i$.

Предимството на този алгоритъм в сравнение с алгоритъма 2 е в това, че тук, както и в алгоритъм 3, не се изисква опериране с големи цели числа и масиви с голяма размерност, но тук трябва да се моделират повече случайни числа, отколкото в алгоритмите 2 и 3.

След известни пресмятания, за трудоемкостта на алгоритъм 4 се получава:

$$S_3 \leq \left(1 + \sum_{i=2}^{n} \frac{N}{N-i+1}\right)(a + b_2) + \sum_{i=2}^{n} \frac{N(i-1)}{N-i+1} c_2 =$$

$$a + b_2 + \sum_{i=2}^{n} \frac{N(a + b_2 + (i-1)c_2)}{N - i + 1}.$$

Не е трудно да се провери,че при големи N и $n \ll N$ алгоритъм 4 е близък по трудоемкост до алгоритми 2 и 3.

2.2.1 Моделиране на събития

С моделиране на дискретни случайни величини с краен брой стойности е свързан и въпросът за моделиране на едно или няколко (съвместни или несъвместни, зависими или независими) случайни събития.

В случая, когато се моделира едно събитие A, което настъпва със зададена вероятност $\Pr(A) = p$, може да се разглежда случайна величина ξ, наречена *индикатор* на събитието A, която е равна на 1 при настъпване на събитието A (с вероятност p), и равна на 0 при настъпване на противоположното събитие $\overline{A}$ (с вероятност $1 - p$)). Тогава алгоритъмът за моделиране на събитието A)се свежда до реализация на стандартно случайно число α и проверка на неравенството $\alpha < p$. Ако това неравенство е изпълнено, то събитието A е настъпило в този опит, а ако $\alpha \geq p$, то не е настъпило.

В случая, когато се моделира пълна група независими по двойки събития $A_1, \ldots, A_n$ (т.е., $A_1 + \cdots + A_n$ е достоверно събитие и $A_i A_j = \emptyset$ при $i \neq j$; знакът за умножение обозначава пресичане на събитията) и са зададени вероятностите $P(A_i) = p_i$ (очевидно, $p_1 + \ldots + p_n = 1$), разглеждаме случайна величина ξ - номер на настъпилото събитие (приема стойност i с вероятност p_i). За осъществяването на всеки опит, генерираме стандартно случайно число α и по алгоритъм 1 определяме стойността на ξ. Ако $\xi = i$, то е настъпило събитието A_i.

В случая, когато се моделира двойка независими съвместни събития A и B ($AB \neq \emptyset$) със зададени вероятности $\Pr(A) = p_A$ и $\Pr(B) = p_B$, те се моделират последователно (поради тяхната независимост): по случайно число α_1 определяме настъпило ли е събитието A, след което по случайно число α_2 определяме дали е настъпило събитието B. По-икономичен е един друг начин. Разглеждаме пълната група взаимно независими събития:

$$A_1 = AB,\ A_2 = A\overline{B},\ A_3 = \overline{A}B,\ A_4 = \overline{AB}, \tag{2.2.8}$$

чиито вероятности не е трудно да се изчислят:

$$p_1 = p_A p_B,\ p_2 = p_A(1 - p_B),\ p_3 = (1 - p_A)p_B,\ p_4 = (1 - p_A)(1 - p_B).$$

Описаният по-горе метод за моделиране на пълна група взаимно независими събития позволява определянето кой от четирите изхода на (2.2.8) е настъпил при поредния опит.

В случая, когато се моделира двойка зависими съвместни събития A и B и са зададени вероятностите p_A, p_B и p_{AB}, може отново да се разгледа пълната група събития (2.2.8), само че вероятностите на тези събития са:

$$p_1 = p_{AB},\ p_2 = p_A - p_{AB},\ p_3 = p_B - p_{AB},\ p_4 = 1 - p_A - p_B + p_{AB}.$$

Да отбележим, че и в този случай е възможно последователното моделиране на събитията A и B, като се използват две стандартни случайни числа. Отначало по случайното число α_1 и вероятността p_A определяме дали е настъпило събитието A (проверка на неравенството $\alpha_1 < p_A$). Ако A е настъпило, то проверката за събитието B се осъществява чрез случайно число α_2 и вероятността $\Pr(B/A) = p_{AB}/p_A$ (проверка на неравенството $\alpha_2 < \Pr(B/A)$). Ако събитието A не е настъпило, то проверката за B се осъществява по случайно число α_2 и вероятността $\Pr(B/A) = (p_B - p_{AB})/(1 - p_A)$.

2.2.2 Моделиране на целочислени случайни величини

Разглеждаме алгоритъм за моделиране на дискретна случайна величина ξ с безкрайно много стойности $x_i, i = 0, 1, 2, \ldots$, и разпределение $\Pr(\xi = x_i) = p_i$. Някои особености на този алгоритъм ще дискутираме върху примера на *целочислени случайни величини*, за които $x_i = i$ и вероятностите p_i са свързани с рекурентни формули от вида $p_{i+1} = p_i r(i)$.

Алгоритъм 6. Стойностите на случайната величина ξ симулираме по следната проста схема: Генерираме стойност на стандартна случайна величина $Q := \alpha$ и полагаме $m := 0$ и $P := p_0$. Извършваме присвояването

$$Q := Q - P \tag{2.2.9}$$

Ако новото Q не е положително, то за стойност на ξ избираме текущото m; в противен случай преизчисляваме вероятността $P := P.r(m)$ и извършваме присвояване $m := m + 1$ и (2.2.9), и отново правим проверка за положителност на Q, и т.н.

Да пресметнем изчислителната сложност на горния алгоритъм - включваме операциите a за моделиране на една стандартна случайна величина,

сравненията на Q с 0 (операциите за едно сравнение означаваме с b_3) и пресмятането на вероятностите (c_3). Получаваме

$$S = a - c_3 + \left(\sum_{i=0}^{\infty}(i+1)p_i\right)(b_3 + c_3) = a - c_3 + \left(1 + \sum_{i=0}^{\infty} ip_i\right)(b_3 + c_3).$$

За целочислени случайни величини

$$\sum_{i=0}^{\infty} ip_i = \mathrm{E}[\xi], \quad \text{и} \quad S = a - c_3 + (1 + E\xi)(b_3 + c_3).$$

Пример 2.2.1 Геометрично разпределение с параметър p (тук и по-нататък, $0 < p < 1$)

$$p_i = p(1-p)^i, \ \ i = 0, 1, 2, \ldots,$$

и в този случай,

$$r(i) = \frac{p_{i+1}}{p_i} = 1 - p, \ \ S = a - c_3 + \frac{(p+1)(b_3 + c_3)}{2}.$$

Пример 2.2.2 Биномно разпределение с параметри p и n (n – естествено число)

$$p_i = C_n^i p_i (1-p)^{n-i}; \ i = 0, 1, 2, \ldots, n; \ C_n^i = \frac{n!}{i!(n-i)!},$$

и в този случай,

$$r(i) = \frac{n!}{(i+1)!(n-i-1)!}\frac{i!(n-i)!}{n!}\frac{p}{1-p} = \frac{n-i}{i+1}\frac{p}{1-p};$$

$$S = a - c_3 + (1 + np)(b_3 + c_3).$$

Пример 2.2.3 Разпределение на Поасон с параметър λ (λ – положително реално число)

$$p_i = \frac{\lambda^i}{i!}e^{-\lambda}; \ i = 0, 1, 2, \ldots,$$

и в този случай,

$$r(i) = \frac{\lambda}{i+1}; \ S = a - c_3 + (1 + \lambda)(b_3 + c_3).$$

Пример 2.2.4 Отрицателно биномно разпределение с параметри p и m (m – естествено число)

$$p_i = C^i_{m+i-1} p^m (1-p)^i;\ i = 0, 1, 2, \ldots,$$

и в този случай,

$$r(i) = \frac{(m+i)(1-p)}{i+1},\ S = a - c_3 + \left(1 + \frac{m(1-p)}{p}\right)(b_3 + c_3).$$

Пример 2.2.5 Разпределение на Паскал с параметри p и m (m – естествено число)

$$p_i = C^{i-m}_{i-1} p^m (1-p)^{i-m};\ i = m, m+1, m+2, \ldots,$$

и в този случай,

$$r(i) = \frac{i(1-p)}{i+1-m},\ S = a - c_3 + \left(1 + \frac{m}{p}\right)(b_3 + c_3).$$

Пример 2.2.6 Логаритмично разпределение с параметър p

$$p_i = -\frac{(1-p)^i}{i \ln p};\ i = 1, 2, \ldots,$$

и в този случай

$$r(i) = \frac{i(1-p)}{i+1},\ S = a - c_3 + \left(1 - \frac{1-p}{p \ln p}\right)(b_3 + c_3).$$

Да отбележим, че шестте примера на дискретни разпределения са свързани с *опитите на Бернули*, т.е. с реализация на стойностите на случайна величина γ с *разпределение на Бернули*:

$$\Pr(\gamma = 1) = p,\ \Pr(\gamma = 0) = 1 - p.$$

Събитието $\{\gamma = 1\}$ се нарича *успех*, а събитието $\{\gamma = 0\}$ – *неуспех*.

Стойността i на случайна величина с геометрично разпределение е броя опити на Бернули до появата на първия успех. Стойността i на случайна величина с биномно разпределение е броят на успехите в n независими опита на Бернули. Тази случайна величина може да се разглежда като сума на n независими опита на Бернули. Разпределението на Поасон с параметър λ е гранична форма на биномното разпределение при

$n \to \infty, p \to 0, np \to \lambda$. Отрицателното биномно разпределение с параметър m описва броя на неуспехите, проявили се преди m-тия успех в серия опити на Бернули. Разпределението на Паскал с параметър m описва броя опити на Бернули, необходими, за да получим m успеха. Логаритмичното разпределение е гранично за отрицателното биномно разпределение с параметър m в смисъл, че

$$\lim_{m \to 0} \Pr(\eta_m = i \mid \eta_m > 0) = -\frac{(1-p)^i}{i \ln p},$$

където η_m е случайна величина с отрицателно биномно разпределение; параметърът m се разглежда като положително реално число, а величината C_{m+i-1} се определя по следния начин:

$$C^i_{m+i-1} = \frac{(m+i-1)(m+i-2)\dots m}{i!}.$$

Тези разсъждения дават още един начин за моделиране на случайните величини от първите пет примера – чрез реализация на съответната серия опити на Бернули. Този начин, обаче, е числено неефективен, поради необходимостта от симулация на голям брой стандартни случайни величини (и.е., равномерно разпределени в интервала $[0, 1]$).

Съществуват различни техники за намаляване на трудоемкостта (изчислителната сложност), като подреждане на вероятностите по реда на тяхното намаляване и използване на специални техники за моделиране.

2.3 Симулиране на непрекъснати случайни величини

2.3.1 Симулиране на непрекъснати случайни величини по метода на обратната функция

Алгоритъм 1. Стандартният метод за моделиране на непрекъсната случайна величина ξ е методът на обратната функция на разпределение, който се изразява чрез:

$$\xi = F^{-1}_{\xi}(\alpha), \tag{2.3.1}$$

където $F^{-1}_{\xi}(.)$ е монотонно растящата функция на разпределение на случайната величина ξ, а α е стандартно случайно число.

Обосноваването на горния алгоритъм е елементарно:

$$\Pr(F^{-1}_{\xi}(\alpha) < x) = \Pr(\alpha < F_{\xi}(x)) = F_{\xi}(x). \tag{2.3.2}$$

Ако функцията $F_\xi^{-1}(.)$ не е строго растяща. то може да се дефинира функцията

$$G_\xi(u) = \inf_{\{x:u<F_\xi(x)\}} x$$

и вместо (2.3.1) разглеждаме моделиращата функция

$$\xi = G_\xi(\alpha); \tag{2.3.3}$$

обоснованието за този алгоритъм съвпада с (2.3.2) с точност да замяната на функцията $F_\xi^{-1}(.)$ със $G_\xi(.)$.

В разглежданите по-нататък примери случайната величина ξ се задава обикновено с плътността на разпределението $p_\xi(.)$. Формулата (2.3.1) се получава, като се реши уравнението

$$\int_{-\infty}^{\xi} p_\xi(u)du = \alpha \tag{2.3.4}$$

относно ξ. Това не винаги е възможно да се направи аналитично, което ограничава приложимостта на този метод.

Пример 2.3.1 *Симулиране на движението на малки инертни частици в среда, състояща се от по-големи частици, които могат да се разсейват или да поглъщат малките частици.*
За случая на еднородна среда, дължината на свободния пробег на малката частица е случайна величина ξ, разпределена по експоненциален закон с плътност, [33]:

$$p_\xi(u) = \lambda e^{-\lambda u},\ u > 0;$$

където параметърът λ е положителен. Функцията на разпределение е

$$F_\xi = \int_0^x p_\xi(u)du = 1 - e^{-\lambda x},$$

и за да се получи формулата за симулация (2.3.2), трябва да се реши уравнението

$$1 - e^{-\lambda\xi} = \alpha$$

относно ξ. Получаваме

$$\xi = -\frac{1}{\lambda}\ln(1-\alpha). \tag{2.3.5}$$

Да отбележим, че величините α и $1-\alpha$ са еднакво разпределени. Наистина,

$$\Pr(1-\alpha < x) = \Pr(\alpha > 1-x) = 1-(1-x) = x.$$

По тази причина, формулата (2.3.5) е по-удобно да се запише във вида

$$\xi = -\frac{1}{\lambda}\ln\alpha.$$

Пример 2.3.2 *Симулиране движението на малка частица в ограничена област без изход от областта (т.нар. блуждаене без изход).*
Движението се свежда до реализация на случайна величина с плътност на разпределение

$$p_\eta = \frac{\lambda e^{-\lambda u}}{1-e^{-\lambda l}},\ 0 \le u \le l, \tag{2.3.6}$$

където l е фиксирано положително число. Разпределението (2.3.6) се нарича *отсечено експоненциално разпределение.*

Твърдение 2.3.1 *Случайната величина с плътност на разпределение* (2.3.6) *може да се симулира с помощта на коя да е от следните четири формули:*

$$\eta = -\frac{1}{\lambda}\ln(1-\alpha(1-e^{-\alpha l})); \tag{2.3.7}$$

$$\eta = -\frac{1}{\lambda}\ln(\alpha+(1-\alpha)e^{-\lambda l}); \tag{2.3.8}$$

$$\eta = t(\alpha) = -\frac{1}{\lambda}\ln\alpha, \quad ако\ 0 \le t(\alpha) \le l; \tag{2.3.9}$$

$$\eta = l\left\{-\frac{1}{\lambda l}\ln\alpha\right\}. \tag{2.3.10}$$

Доказателство. Формулата (2.3.7) е решение на уравнението (2.3.4), а формулата (2.3.8) се получава при замяна на α с $1-\alpha$ в (2.3.7).

Да намерим функцията на разпределение на случайната величина (2.3.9):

$$\Pr(\eta < x) = \Pr(t(\alpha) < x \mid 0 \le t(\alpha) \le l)$$

при $0 < x < l$. Съгласно дефиницията на условна вероятност,

$$\Pr(\eta < x) = \frac{\Pr((t(\alpha) < x) \cap (0 \le t(\alpha) \le l))}{\Pr(0 \le t(\alpha) \le l))} = \frac{\Pr(t(\alpha) < x)}{\Pr(0 \le t(\alpha) \le l)} =$$

$$\frac{\Pr(-\lambda x < \ln\alpha)}{\Pr(-\lambda l \le \ln\alpha \le 0)} = \frac{\Pr(e^{-\lambda x} < \alpha)}{\Pr(e^{-\lambda l} \le \alpha \le 1)} = \frac{1-e^{-\lambda x}}{1-e^{-\lambda l}},$$

последният израз представлява функцията на разпределение на случайната величина с плътност (2.3.6).

Да получим и функцията на разпределение на случайната величина (2.3.10):

$$\Pr(\eta < x) = \Pr\left(\left\{-\frac{1}{\lambda l}\ln\alpha\right\} < \frac{x}{l}\right) = \sum_{k=0}^{\infty}\Pr\left(k \le -\frac{1}{\lambda l}\ln\alpha < k + \frac{x}{l}\right) =$$

$$= \sum_{k=0}^{\infty}\Pr(-kl\lambda - x\lambda < \ln\alpha \le -kl\lambda) = \sum_{k=0}^{\infty}\Pr(e^{-kl\lambda - x\lambda} < \alpha < e^{-kl\lambda}) =$$

$$= \sum_{k=0}^{\infty}(e^{-kl\lambda} - e^{-kl\lambda - x\lambda}) = \sum_{k=0}^{\infty} e^{-kl\lambda}(1 - e^{-\lambda x}) = \frac{1 - e^{-\lambda x}}{1 - e^{-\lambda l}}.$$

Тук се използва обстоятелството, че сумата на членовете на безкрайна геометрична прогресия с първи член 1 и знаменател $e^{-\lambda l}$ е равна на $1/(1 - \exp(-\lambda l))$.

2.3.2 Симулиране на непрекъсната случайна величина с метод на селекцията

Този метод се използва, когато не може да се приложи метода на обратната функция. Искаме да симулираме случайна величина с плътност f. Предполагаме, че разполагаме с метод за симулиране на случайна величина с плътност g, при което задачата ни се свежда до използване на извадки от g за симулиране с дадената плътност f. Функцията g може да бъде произволна, освен изискването да изпълнява едно условие, което е споменато по-долу. Основната идея на метода на селекцията е да се симулира разпределение с плътност g и получените реализации да се приемат за реализации на желаното разпределение с вероятност, пропорционална на съотношението f/g. Нека C е константа, такава че

$$\frac{f(Y)}{g(Y)} \le C, \quad \text{за всяко} \quad Y.$$

Тогава, процедурата за симулация е:

(а) Симулираме сл. в. Y с плътност g и симулираме U равномерно разпределена в $[0, 1]$.

(б) Ако $U \le \frac{f(Y)}{g(Y)C}$, то $X = Y$, иначе отиваме на стъпка (а).

ределение на случай-

случайната величина

$\cdot\frac{1}{\lambda l}\ln\alpha < k + \frac{x}{l}\Big) =$

$^{-x\lambda} < \alpha < e^{-kl\lambda}) =$

$= \frac{1-e^{-\lambda x}}{1-e^{-\lambda l}}.$

ıленовете на безкрай-
ıател $e^{-\lambda l}$ е равна на

на величина с ме-

ıжи метода на обрат-
ıичина с плътност f.
не на случайна вели-
да до използване на
Функцията g може да
едно условие, което е
екцията е да се симу-
ıзации да се приемат
ıост, пропорционална
е

ираме U равномерно

стъпка (а).

Твърдение 2.3.2 *X е случайна величина с плътност f.*

Доказателство. Нека X е получената стойност и n е броят на итерациите, необходими за достигане на тази стойност. Тогава,

$$\begin{aligned}\Pr(X \le x) &= \Pr(Y_n \le x)\\ &= \Pr\left(Y \le x \mid U \le \frac{f(Y_n)}{g(Y_n)C}\right)\\ &= \frac{\Pr\left(Y \le x, U \le \frac{f(Y_n)}{Cg(Y_n)}\right)}{K},\end{aligned}$$

където $K = \Pr\left(U \le \frac{f(Y_n)}{Cg(Y_n)}\right)$. Тъй като Y и U са независими случайни величини, то тяхната съвместна плътностна функция е произведение от маргиналите

$$g(y) \times 1,$$

тъй като U е равномерна разпределена случайна величина в $[0,1]$. Следователно,

$$\begin{aligned}P(X \le x) &= \frac{1}{K}\int_{\frac{U<f(Y)}{Cg(Y)}}\int_{Y\le x} g(Y)dY\,dU\\ &= \frac{1}{K}\int_{-\infty}^{x}\left(\int_0^{\frac{f(Y)}{Cg(Y)}} dU\right) g(Y)dY\\ &= \frac{1}{K}\int_{-\infty}^{x}\frac{f(Y)}{Cg(Y)}g(Y)dY\\ &= \frac{1}{CK}\int_{-\infty}^{x} f(Y)dY\end{aligned}$$

За $x \to \infty$ лявата страна клони към 1 и интеграла от дясната страна също клони към 1. Следователно, $CK = 1$ и

$$P(X \le x) = \int_{-\infty}^{x} f(Y)dY.$$

Следователно, X е случайна величина с вероятностна плътност f.

За дадена стойност на Y ние приемаме Y чрез генериране на U и сравнение на U с $\frac{f(Y)}{Cg(Y)}$. Казваме също, че приемаме Y с вероятност $\frac{f(Y)}{Cg(Y)}$. Всяка итерация на този цикъл включва независими реализации, и ние пресмятаме вероятността за приемане на Y като X в съответствие с

$$\Pr\left(U \leq \frac{f(Y)}{Cg(Y)}\right) = K = \frac{1}{C}.$$

Ако C е голямо, то процесът на генериране на извадки от f с използване на този метод ще бъде бавен.

Сега ще илюстрираме използването на метода на селекцията за генериране на извадка на случайна величина със стандартно нормално разпределение. Като първа стъпка ще симулираме с плътностната функция:

$$f(x) = \frac{2}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right), \quad x \geq 0.$$

Да отбележим, че това е плътност, асоциирана с $x = |z|$, където z е стандартна нормална случайна величина. Освен това, предполагаме, че имаме средства за получаване на извадка със стандартна експоненциална плътност, която се използва като g от горния алгоритъм ($g(x) = \exp(-x)$). Да ограничим отношението на f и g с константа:

$$\frac{f(x)}{g(x)} = \sqrt{2\pi}\ \exp\left(-\frac{(x^2-2x)}{2}\right)$$

$$= \sqrt{\frac{2e}{\pi}}\ \exp\left(-\frac{(x-1)^2}{2}\right)$$

$$\leq \sqrt{\frac{2e}{\pi}} = C \qquad .$$

и

$$\frac{f(x)}{Cg(x)} = \exp\left(-\frac{(x-1)^2}{2}\right).$$

Използваме следния алгоритъм за генериране на случайна величина с плътност f:

1. Генериране на Y, експоненциална случайна величина с очакване 1, и U, равномерно разпределена в $[0,1]$ случайна величина.

2. Ако $U \leq \exp\left(\frac{-(Y-1)^2}{2}\right)$ полагаме $X = Y$, иначе се връщаме на (1).

Сега,

$$U \leq \exp\left(\frac{-(Y-1)^2}{2}\right)$$

$$-\log(U) \geq \frac{-(Y-1)^2}{2}$$

Тъй като $-\log(U)$ е експоненциална със скорост 1, реформулираме по следния начин:

1. Генерираме Y_1 и Y_2, извадки от експоненциална случайна величина с очакване 1.

2. Ако $Y_2 \geq \frac{-(Y_2-1)^2}{2}$ полагаме $X = Y_1$, иначе се връщаме на (1).

След като сме генерирали случайна величина, която е абсолютна стойност на стандартно нормално разпределена, можем да генерираме извадка от стандартно нормално разпределение съгласно следния алгоритъм.

1. Генерираме U, равномерно разпределена в $[0,1]$ случайна величина, и генерираме X по гореописания алгоритъм.

2. Ако $U \in \left(0, \frac{1}{2}\right]$ полагаме $Z = X$, иначе полагаме $Z = -X$.

Полярен метод: Известен е един друг метод за генериране на извадки от стандартното нормално разпределение. Нарича се метод на **Бокс-Мюлер (Box-Muller)**. Принципът зад този метод е:

Ако X и Y са независими и стандартни нормални случайни величини, то за

$$\theta = \tan^{-1}\left(\frac{Y}{X}\right), \;\; R = \sqrt{(X^2+Y^2)}$$

θ е равномерна в $[0, 2\pi]$ и R^2 е експоненциална с очакване 2.

Обръщайки този резултат, ако U_1 и U_2 са равномерни в $[0,1]$, то за

$$R = (-2\log(U_1))^{1/2}, \;\; \theta = 2\pi U_2,$$

и

$$X = R\cos(\theta), \;\; Y = R\sin(\theta)$$

X и Y са независими извадки с плътност на стандартно нормално разпределение. Недостатък е използването на тригонометрични функции, които винаги са изчислително скъпи. Този метод може да се модифицира по следния начин, за да се избегнат синусите и косинусите.

Ако U_1 и U_2 са равномерни в $[0,1]$, то

$$V_1 = 2U_1 - 1, \;\; V_2 = 2U_2 - 1,$$

са равномерни в $[-1,1]$. Те могат да лежат или да не лежат в кръга с радиус 1 и център в (0,0). Генерираме двойката (V_1, V_2) докато тя лежи в кръга и нека $(\widehat{R}, \widehat{\theta})$ са полярните координати на тази двойка. Може да се покаже, че $\widehat{R}^2$ е равномерна в $[0,1]$ и $\widehat{\theta}$ е равномерна в $[0, 2\pi]$. За тези стойности

$$\sin(\widehat{\theta}) = \frac{V_2}{R}, \ \cos(\widehat{\theta}) = \frac{V_1}{R},$$

Следователно, ако U е равномерно разпределена в $[0,1]$ случайна величина, независима от $\widehat{\theta}$, то

$$X = (-2\log(U))^{1/2}\frac{V_1}{R}$$

$$Y = (-2\log(U))^{1/2}\frac{V_2}{R}$$

са независима извадка от стандартната нормална плътност. Тъй като $\widehat{R}^2$ е равномерна в $[0,1]$ и независима от $\widehat{\theta}$, то тя може да се използва вместо U. Следователно, X и Y могат да бъдат генерирани съответно на уравненията,

$$X = (-2\log(\widehat{R}^2))^{1/2}\frac{V_1}{R}$$

$$Y = (-2\log(\widehat{R}^2))^{1/2}\frac{V_2}{R}$$

Ето един алгоритъм за използване на тези уравнения за генериране на независими извадки от стандартно нормално разпределение:

1. Генериране на независима равномерна сл. в. в $[0,1]$.

2. Полагаме $V_1 = 2U_1 - 1$, $V_2 = 2U_2 - 1$ и $S = V_1^2 + V_2^2$.

3. Ако $S > 1$, връщаме се към (1). Иначе полагаме $T = \sqrt{\frac{-2\log(S)}{S}}$, $X = TV_1$, $Y = TV_2$.

За да генерираме Y, нормална случайна величина със средно μ и стандартно отклонение σ, генерираме стандартна нормална случайна величина X и полагаме:

$$Y = \sigma X + \mu.$$

2.3.3 Интерполация на вероятностна плътност на разпределение

Разглеждаме задачата за апроксимиране на неизвестна функция на плътност $p(x)$, дефинирана в $[a,b]$, по дадени N реализации $\{\xi\}_{i=1}^{N} \in [a,b]$ на

случайната величина ξ с плътност $p(x)$. В работата [19] е представен Монте Карло метод за решаване на тази задача.

Предполагаме, че $p(x) \in C^k[a,b]$, където $k \geq 0$ е цяло число. Интервалът $[a,b]$ се разделя на m подинтервали и се въвежда следната мрежа от точки:

$$\omega_m = \{a = x_0 < x_1 < \cdots < x_m = b\}$$

със стъпка $h = \frac{b-a}{m}$. Към множеството ω_m се добавят $2k$ нови възли, в резултат на което се получава множеството:

$$T_n = \{t_1 < t_2 < \cdots < t_{k+1} = x_0 < \cdots < x_m = t_{n-k} < t_{n-k+1} < \cdots < t_n\},$$

при което $n = 2k + m + 1$.

Разглеждаме следната апроксимация на неизвестната плътност $p(x)$ с B-сплайни от степен k:

$$p(x) = \sum_{i=1}^{L} c_i B_{i,k}(x),\ x \in [a,b],\ L = n - k - 1$$

с грешка $O(h^k)$, където c_i, $i = 1, \ldots, L$ са апроксимационните коефициенти.

i-тият B-сплайн от степен k с възли $t_i, \ldots, t_{i+k+1}$ се дефинира като разделена разлика на отсечената степенна функция $(t-x)_+^k$ по отношение на t $t_i, \ldots, t_{i+k+1}$:

$$B_{i,k}(x) = (\cdot - x)_+^k[t_i, \ldots, t_{i+k+1}],\ i = 1, \ldots, L,$$

където x е фиксирана точка и $(t-x)_+^k = (t-x)^k$ за $t > x$, $(t-x)_+^k = (t-x)^k$ при $t \leq x$. Следователно, $B_{i,k}(x)$ се представя в следния вид:

$$B_{i,k}(x) = \sum_{s=i}^{i+k+1} \frac{(t_s - x)_+^k}{\omega'_{i,k}(t_s)},$$

където $\omega_{i,k}(t) = (t - t_i) \cdots (t - t_{i+k+1})$.

Предимството при прилагането на сплайни за апроксимация на неизвестна функция е в използването на алгебрични полиноми от ниска степен. Повишаването на точността при приближаването на функцията може да се постигне чрез разделяне на малки подинтервали.

За да получим коефициентите c_i $(i = 1, \ldots, L)$, прилагаме метода на най-малките квадрати, т.е., коефициентите на апроксимацията се избират така, че да се минимизира стойността на интеграла от квадрата на грешката, т.е., функцията

$$U = \int_a^b \left(p(x) - \sum_{i=1}^{L} c_i B_{i,k}(x) \right)^2 dx$$

да приема най-малка стойност. Функцията U е функция на L променливи:

$$U = U(c_1, \ldots, c_L) = \int_a^b (p(x) - \varphi(x; c_1, \ldots, c_L))^2 dx$$

$$U = \int_a^b p^2(x)dx - 2\sum_{i=1}^{L} c_i(p, B_{i,k}(x)) + \int_a^b \left(\sum_{i=1}^{L} c_i B_{i,k})(x) \right)^2 dx.$$

Функцията $U = U(c_1, \ldots, c_L)$ има непрекъснати частни производни по всички променливи. Необходимото условие функцията да притежава минимум се изразява със следната система от линейни алгебрични уравнения:

$$\frac{\partial U}{\partial c_1} = 0, \quad \frac{\partial U}{\partial c_2} = 0, \quad \ldots, \quad \frac{\partial U}{\partial c_L} = 0.$$

Получаваме

$$\frac{\partial U}{\partial c_i} = -2(p, B_{i,k}) + 2\int_a^b \left(\sum_{j=1}^{L} c_j B_{j,k}(x) \right) B_{i,k}(x)dx = 0$$

$$\sum_{j=1}^{L} (B_{i,k}(x), B_{j,k}(x))c_j = (p(x), B_{i,k}(x)), \quad i = 1, \ldots, n-k-1,$$

където скаларното произведение се дефинира по следния начин:

$$(f, g) = \int_a^b f(x)g(x)dx.$$

Търсените апроксимационни коефициенти c_j се получават като решение на системата с L уравнения. Да отбележим, че в дясната страна на системата са скаларните произведения на неизвестната функция $(p(x))$ с известни функции (B-сплайните). От друга страна, скаларното произведение

$(p(x), B_{i,k}(x))$ е математическото очакване на случайната величина $B_{i,k}(x)$ с плътност на разпределение $p(x)$:

$$(p(x), B_{i,k}(x)) = \int_a^b p(x) B_{i,k}(x) dx = E B_{i,k}(\xi),$$

където случайната точка ξ има плъност $p(x)$. Скаларното произведение $(p(x), B_{i,k}(x))$ се оценява като се използва Монте Карло метод за пресмятане на интеграли (виж Глава 4 от книгата):

$$(p(x), B_{i,k}(x)) \approx \frac{1}{N} \sum_{j=1}^{N} B(\xi_j) = \widehat{\theta_N},$$

където $\{\xi_i\}_{i=1}^N$ е дадената извадка. Всяка B-сплайн функция от степен k се дефинира единствено в краен брой възли $k+2$. Така първият и последният възел могат да бъдат разглеждани като граници на подинтервал, съответстващ на подинтервал, получен чрез разделяне на областта.

2.3.4 Оценяване на вероятностна плътност с хистограми

Хистограмите представляват графичен начин за сумиране или описване на данни. Хистограмите показват визуално разпределението на дадено множество от данни и дават информация за относителните честоти на наблюденията. Хистограмите са най-старият и широко използван непараметричен метод за оценка на плътността. Обикновено хистограмата се формира чрез разделяне на реалната права на интервали с равни дължини, често наричани стълбове, и като се използва случайна извадка $\xi_1, \xi_2, \ldots, \xi_N$. Първо се избира началото x_0 на стълбовете и ширината h на стълба. Ширината h на стълба обикновено се нарича изглаждащ параметър, тъй като контролира изглаждането, приложено към данните. Целта е да се оцени функцията на вероятностна плътност, т.е., да се получи функция $\widehat{p(x)}$, която е неотрицателна и удовлетворява следното условие:

$$\int_a^b \widehat{p(x)} dx = 1.$$

Оценката на едномерната хистограма $\widehat{p_{Hist}}(x)$ в точка x се дефинира със следния израз:

$$\widehat{p_{Hist}}(x) = \frac{\nu_i}{hN} = \frac{1}{hN} \sum_{i=1}^{N} I_{S_j}(\xi_i), \;\; x \in S_j,$$

където $S_j = [x_j, x_{j+1})$ е j-тият стълб $(j = 0, \ldots, m-1)$, а ν_j е броят на реализациите (опитите) в j-тия стълб $\left(\sum_{j=0}^{m-1} \nu_j = N\right)$ и $I_{S_j}(\xi_i)$ е характеристичната функция на стълба S_j.

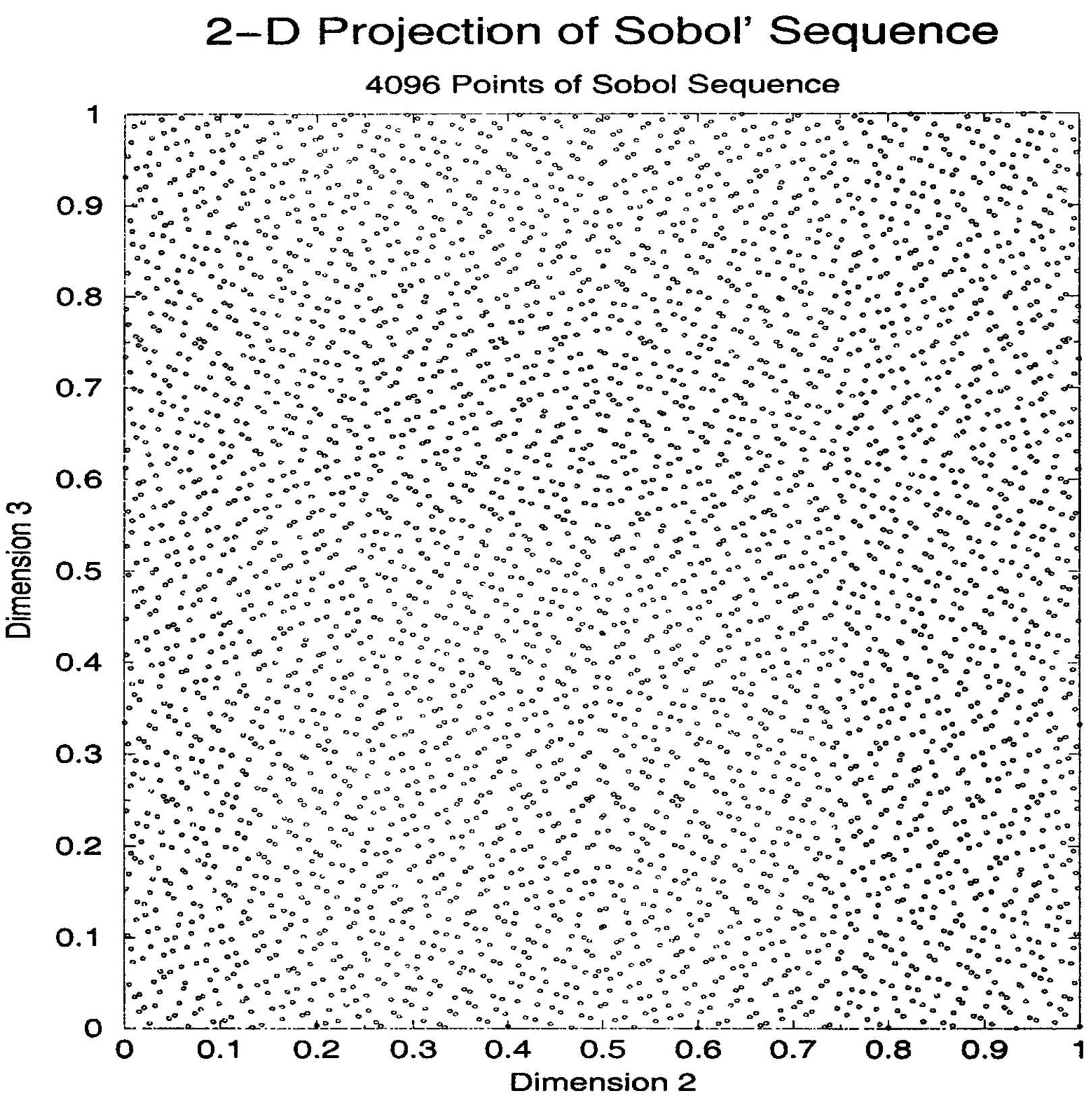
Глава 3

Квазислучайни редици

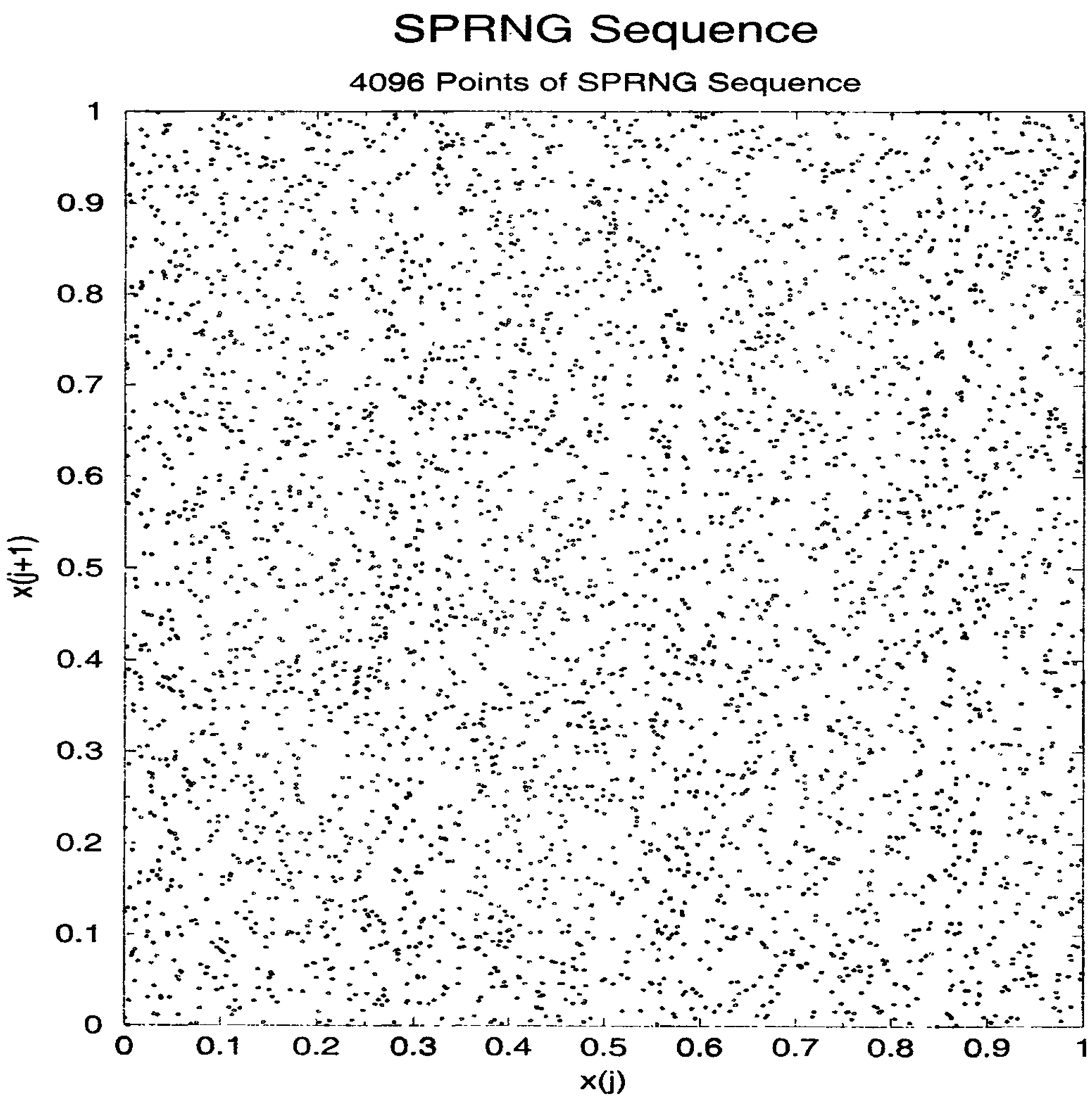
Монте Карло методите се основават на псевдослучайни числа, а квази-Монте Карло методите се основават на детерминистични редици с малък дискрепанс (наричани квазислучайни редици). Докато псевдослучайните генератори са конструирани така, че да имитират поведението на истински случайните числа, квазислучайните редици са построени така, че да са толкова равномерно разпределени, колкото е математически възможно (Фигура 3.1). При използване на квазислучайни редици, сходимостта на квази-Монте Карло методите е $O(N^{-1}\log^s N)$ (Фигура 3.2).

Фигури 3.1 представя една от двумерните проекции на 28-мерна квазислучайна редица на Собол а Фигура 3.2 визуализира 2-мерно разпределение на псевдослучайни числа, получени чрез генератор от библиотека SPRNG (`http://sprng.cs.fsu.edu/`)

Квазислучайните редици, наричани понякога редици с малък дискрепанс, обикновено е генерирана в единичния s-мерен хиперкуб, $I_s = [0,1]^s$. Първоначалното конструиране на квазислучайните редици е свързано с редицата на ван дер Корпут (van der Corput), която е едномерна квазислучайна редица, основана на цифрова инверсия. Понастоящем, съществуват много квазислучайни редици с различни основи и размерности, основани на метода на инверсията, напримерр, редиците на Холтън (1960 г.), който генерализира редицата на ван дер Корпут до произволна размерност, и редиците на Собол (1967, 1976 г.). Съществено обобщение на метода на инверсията прави Фор (Faure) и през 1982 г. конструира редица, която днес носи неговото име. По-късно, Нидерайтер (Niederreiter) обобщава известните конструкции на Собол и Фор и създава редици с произволни основи, известни като редиците на Нидерайтер (1992 г.). По-нататък,



Фигура 3.1: Двумерна проекция на редица на Собол



Фигура 3.2: Двумерна псевдослучайна редица, генерирана от библиотеката **SPRNG**

Тезука [1993 г.] обобщава редиците на Нидерайтер чрез използване на полиномиален аналог на редиците на Холтън. Нови редици се свързват с имената на Xing и Niedereiter (1995). Owen (1995, 1997, 1998), Hickernell (1996). Всички тези редици удовлетворяват оценката от дефиниция 3.1.2 (дадена по-долу), но по-късно създадените редици имат по-малка константа. От фигурите 3.1 и 3.2 се вижда, че псевдослучайните числа клонят към групиране в клъстери, докато квазислучайните са равномерно разпределени.

Мярката за липса на равномерност (или еднаква разпределеност) на точки разположени в едно множество, най-често единичния хиперкуб, $I_s = [0,1]^s$, се нарича ***дискрепанс***. Най-широко изследваните мерки за дискрепанс се основават на L_p норма; при $p = 2$ дискрепанса се нарича L_2-дискрепанс, при $p = \infty$ се нарича дискрепанс-звезда и има следната дефиниция [46]:

Дефиниция 3.0.1 *За всяка редица* $\{x_n\} \in [0,1)_s$ *с* N *елемента,* $x_i = (x_i^{(1)}, x_i^{(2)}, \ldots x_i^{(s)})$, *при* $J(\nu) = [0,\nu_1) \times [0,\nu_2) \times \ldots \times [0,\nu_s)$, *дискрепанс-звезда на редицата,* $D_N^\star$, *се дефинира като*

$$D_N^\star = \sup_{0 \le \nu_j < 1} \left| \frac{1}{N} \{x_i \in J(\nu)\} - \prod_{j=1}^{s} \nu_j \right|, \tag{3.0.1}$$

където $0 \le \nu_j \le 1$.

Конструирането на квазислучайните редици се основава на минимизация на техния дискрепанс. Дефиницията на квазислучайна редица, основана на звезда-дискрепанса е следната:

Дефиниция 3.0.2 *За всяко* $N > 1$ *означаваме с* $\{x_i\}_{1 \le i \le N}$ *първите* N *точки от редицата* $\{x_i\}$. *Казваме, че редицата* $\{x_i\}$ *е редица с малък дискрепанс, ако е изпълнено*

$$D_N^\star \le C_s \frac{(logN)^s}{N}, \tag{3.0.2}$$

където константата C_s *зависи само от размерността* s.

3.1 Квазислучайни генератори

3.1.1 Редици, основани на ирационални числа

Един от най-старите класове от редици с малък дискрепанс се получава при разглеждане на кратни на подходящи ирационални вектори по модул 1. За $u \in \mathbb{R}$ нека $\{u\} = u - \lfloor u \rfloor$ е дробната част на u. Очевидно, $\{u\} \in [0,1)$. Дефинираме дробната част на $u = (u_1, \ldots, u_s) \in \mathbb{R}^s$ чрез

$$\{u\} = (\{u_1\}, \ldots, \{u_s\}) \in [0,1)^s.$$

Тогава, ако $\alpha = (\alpha_1, \ldots, \alpha_s) \in \mathbb{R}^s$ е такова, че $1, \alpha_1, \ldots, \alpha_s$ са линейни независими върху множеството на рационалните числа, редицата

$$x_n = \{n\alpha\}, \text{ за всяко } n > 0$$

е равномерно разпределена в $[0,1]^s$ съгласно класическия резултат на Weyl (1916 г.).

Дефинираната по-горе редица е известна в литературата под много имена. Някои изследователи я свързват с един от първите учени, изследвали такъв тип редица, и я наричат „редица на Кронекер“, други използват името „редица на Вейл“ или „редица на Ричмаер“.

3.1.2 Редица на Холтън

Нека $b \geq 2$ е цяло число, тогава всяко цяло $n \geq 0$ има единствено представяне:

$$n = \sum_{j=0}^{\infty} d_j b^j \tag{3.1.1}$$

при основа b, където $d_j \in \mathbb{Z}_b$ за всички $j \geq 0$ и $d_j = 0$ за достатъчно голямо j.

Дефиниция 3.1.1 *За всяко цяло число $b \geq 2$, обръщащата (radical-inverse) функция Φ_b при основа b се дефинира като*

$$\Phi_b(n) = \sum_{j=0}^{\infty} d_j b^{-j-1} \qquad n \geq 0, \tag{3.1.2}$$

където n е дадено чрез цифрово представяне (3.1.1) ***при*** *основа* b.

Така $\Phi_b(n)$ се получава от n чрез симетрично завъртане на представянето (3.1.1) около „десетичната точка“. Да отбележим, че Φ_b е в интервала $[0,1)$ за всяко $n \geq 0$. По-долу е представен алгоритъм за пресмятане на Φ_b; в него $\lfloor x \rfloor$ връща цялата част на x. Алгоритъмът има аритметична сложност $O(\log_b n)$ за пресмятане на n-тата точка.

Алгоритъм за пресмятане на Φ_b

$\Phi_b = 0,\ k = n,\ b' = 1$
while $k > 0$ do
$b' = b'/b$
$a = k \mod b$
$\Phi_b = \Phi_b + ab'$
$k = \lfloor k/b \rfloor$
end while

За цяло число $b \geq 2$ редицата на Ван дер Корпут при основа b е едномерната редица $x_0, x_1, \ldots$, със $x_n = \Phi_b(n)$ за всяко $n \geq 0$.

Да отбележим, че редицата на Ван дер Корпут при основа b е $(0,1)$-редица при основа b съгласно дефиницията на Нидерайтер, дадена по-нататък. Случаят, когато $b = 2$ е редицата, която Ван дер Корпут публикува през 1937 г.

Холтън разширява дефиницията на редицата на Ван дер Корпут до редица с малък дискрепанс в произволна размерност. За дадена размерност $s \geq 1$ нека $b_1, \ldots b_s$ са цели числа по-големи от 1 и взаимно прости. Тогава, редица на Холтън при основи $b_1, \ldots b_s$, $s \geq 1$ е редицата $x_0, x_1, \ldots$, за която:

$$x_n = (\Phi_{b_1}(n), \ldots, \Phi_{b_n}(n)) \in [0,1)^s \text{ for all } n \geq 0.$$

Използвайки горния алгоритъм, изчислителната стойност за генериране на N точки от редицата на Холтън в размерност s е $O(sN \log_{b_{\min}} N)$ (където $b_{\min} = \min\{b_1, \ldots, b_s\}$.

3.1.3 Точково множество на Хамерсли

Чрез избор на равноотдалечени точки за първата координата, Хамерсли дефинира крайно точково множество от редицата на Холтън. За избора на първата координата има няколко предложения.

Например, Нидерайтер и Шоу предлагат при $s \geq 2$, за цели числа $N \geq 1$ и $b_1, \ldots, b_{s-1} \geq 2$, N-тият елемент на точково множество на Хамерсли при основи $b_1, \ldots, b_{s-1}$ да бъде дефиниран като:

$$x_n = \left(\frac{n}{N}, \Phi_{b_1}(n), \ldots, \Phi_{b_{s-1}}(n)\right) \in [0,1)^s \text{ за } n = 0, 1, \ldots, N-1. \quad (3.1.3)$$

Друга дефиниция с много малка разлика е дадена от Холтън:

$$x_n = \left(\frac{n}{N}, \Phi_{b_1}(n), \ldots, \Phi_{b_{s-1}}(n)\right) \in [0,1)^s \text{ за } n = 1, \ldots, N.$$

Да отбележим, че тук n се променя от 1 до N вместо от 0 до $N-1$, като точковото множество се различава от (3.1.3) само с една точка.

Накрая, Фанг и Уанг използват

$$x_n = \left(\frac{2n-1}{2N}, \Phi_{b_1}(n), \ldots, \Phi_{b_{s-1}}(n)\right) \in [0,1)^s \text{ for } n = 1, \ldots, N.$$

като дефиниция на точково множество на Хамерсли.

3.1.4 (t, m, s)-мрежи и (t, s)-редици

Един друг клас от квазислучайни редици се формира от (t, m, s)-мрежи и (t, s)-редици. Конструкцията на (t, m, s)-мрежи и (t, s)-редици е тясно свързана с понятието *елементарен интервал при основа* b:

Дефиниция 3.1.2 *Елементарен интервал при основа b е интервал E от $[0,1)^s$ във вида:*

$$E = \prod_{i=1}^{s} \left[\frac{a_i}{b^{d_i}}, \frac{a_i + 1}{b^{d_i}}\right),$$

където $a_i, d_i \in \mathbb{Z},\ d_i \geq 0,\ 0 \leq a_i \leq b^{d_i}$ за $1 \leq s$.

По този начин един елементарен интервал E е подинтервал на $[0,1)^s$ чиято j-та ос има дължина $1/b^{d_j}$.

Дефиниция 3.1.3 *Нека $0 \geq t \geq m$ са цели числа. Една (t,m,s)-мрежа при основа b е крайно точково множество P с b^m точки в $[0,1]^s$, такова че всеки елементарен интервал при основа b с обем b^{t-m} съдържа точно b^t точки от P.*

По-малка стойност на t води до мрежа с по-добра равномерност (по-малък дискрепанс). Всяка (t_1,m,s)-мрежа е (t_2,m,s)-мрежа за $t_1 < t_2$ и тривиално може да се провери, че всяко точково множество е (m,m,s)-мрежа.

Дефиниция 3.1.4 *Нека $t \geq 0$ е цяло число. Безкрайната редица от точки $x_0, x_1, \ldots$ в $[0,1)^s$ е (t,s)-редица при основа b, ако за всички цели числа $k \geq 0$ и $m > t$ крайното точково множество, състоящо се от точките x_n, за които $kb^m \leq n < (k+1)b^m$, е (t,m,s)-мрежа при основа b.*

Грубо казано, една (t,s)-редица при основа b има свойството, че последователните точки се разполагат еднакво в елементарните интервали при основа b, определени от кордажа (разделящите прави) на b^m.

3.1.5 Редица на Собол

При представяне по битове на n

$$n = \sum_{j=1}^{w} b_j 2^{j-1},$$

където $b_j \in 0,1$, то j-тата координата на n-тата точка от *редицата на Собол* се генерира чрез:

$$x_n^{(j)} = b_1 v_1^{(j)} \oplus b_2 v_2^{(j)} \oplus \ldots b_w v_w^{(j)},$$

където $v_i^{(j)}$ е двоична дроб, наричана i-то направляващо число на j-тата размерност, и където $\oplus$ означава побитова exclusive-or операция (например, $7/8 \oplus 11/16 = 0.1110 \oplus 0.1011 = 0.\ 0101 = 5/16$). За получаването на $v_i^{(j)}$, за всяка размерност се избира различен примитивен полином (т.е., полином, чиито коефициенти имат за общ делител само 1):

$$P_j(x) = x^{d_j} + a_1^{(j)} x^{d_j - 1} + \ldots + a_{d_j - 1}^{(j)} x + 1$$

от степен d_j над полето GF(2). Да отбележим, че свободният член е равен на 1, защото P_j е примитивен полином; броят на примитивните полиноми от степен d е $\Phi(2^d - 1)/d$, където Φ е тотиентната функция на Ойлер (тотиент на положително цяло число n се дефинира като броя на положителните цели числа, по-малки или равни на n, които са взаимно прости с n. Например, тотиента на 9 е 6, защото има шест цели числа, които са взаимно прости с 9).

След като е избран полином за определена размерност, то неговите коефициенти се използват за дефиниране на d_j-тия член на рекурентната релация:

$$v_i^{(j)} = a_1^{(j)} v_{i-1}^{(j)} \oplus a_2^{(j)} v_{i-2}^{(j)} \oplus \ldots \oplus a_{d_j-1}^{(j)} v_{i-d_j+1}^{(j)} \oplus v_{i-d_j}^{(j)} \oplus \frac{v_{i-d_j}^{(j)}}{2^{d_j}} \quad i > d_j.$$

Последният член е $v_{i-d_j}^{(j)}$, преместен надясно с d_j позиции. Ако дефинираме

$$v_i^{(j)} = m_i^{(j)} / 2^I,$$

където m_i е нечетно цяло число и $0 < m_i < 2^I$, рекурсията може да се представи еквивалентно в термините на m_i като

$$m_i^{(j)} = 2a_1^{(j)} m_{i-1}^{(j)} \oplus 2^2 a_2^{(j)} m_{i-2}^{(j)} \oplus \ldots$$
$$\ldots \oplus 2^{d_j - 1} a_{d_j-1}^{(j)} m_{i-d_j+1}^{(j)} \oplus 2^{d_j} m_{i-d_j}^{(j)} \oplus m_{i-d_j}^{(j)}.$$

Формата на рекурсията е много подходяща за програмно изпълнение, защото лесно може да се програмира с използване на целочислена аритметика. Стойностите на $m_1^{(j)}, m_2^{(j)}, \ldots, m_d^{(j)}$ могат да бъдат избрани свободно при спазване на изискването m_i е нечетно и $m_i^{(j)} < 2^I$.

Освен горната процедура, съществува и по-бърз начин за конструиране на редицата на Собол. В [6] Антонов и Салеев доказват, че представянето

$$x_n^{(j)} = g_1 v_1^{(j)} \oplus g_2 v_2^{(j)} \oplus \ldots$$

където $\ldots g_3 g_2 g_1$ е двоичното представяне на n с код на Грей, не променя асимптотично дискрепанса на редицата. За $k = 2, 3, \ldots$ кодът на Грей

просто размества всеки начален сегмент с дължина 2^k от оригиналната редица на Собол. Двоичният код на Грей има следните свойства:

1. Двоичният код на Грей за n се получава от двоичното представяне на n посредством:

$$\ldots g_3 g_2 g_1 = \ldots b_3 b_2 b_1 \oplus \ldots b_4 b_3 b_2.$$

2. Двоичният код на Грей за n и двоичния код на Грей за $n+1$ се различават само в една позиция. За да се намери тази позиция, трябва да се търси най-десния нулев бит в двоичното представяне на n (добавяне водеща нула към n, ако няма възможност за друго). Ако наречем този бит b_c, то тогава бита, чиято стойност се променя е g_c.

Използвайки тези свойства, $x_{n+1}^{(j)}$ може да се представи чрез $x_n^{(j)}$ по следния начин:

$$x_{n+1}^{(j)} = x_n^{(j)} \oplus v_c^{(j)},$$

където b_c е най-десния нулев бит в двоичното представяне на n. Това прави варианта на Антонов-Салеев много по-бърз от оригиналната схема на Собол. За да започне рекурсията, полагаме $x_0^{(j)} = 0$ за $1 \leq j \leq s$.

Собол е доказал, че ако за горната конструкция всички примитивни полиноми $P_j(x)$ са различни, то получената редица е (t, s)-редица със

$$t = \sum_{j=1}^{s} (\deg(P_j) - 1)$$

при $P_1(x) = x$ и P_{j+1} означава j-тия примитивен полином над $\mathbb{Z}_2$ нареден по степента.

Игнорирайки работата, необходима за пресмятане на началните направляващи числа, изчислителната сложност за генериране на s-мерни точки на Собол изисква първо да се определи c, най-десния нулев бит на n, последвано от побитова exclusive-or операция. Определянето на c изисква $O(\log n)$ операции, и при използване на побитова exclusive-or операция, то генерирането на s-мерен вектор изисква време $O(n + \log n)$. Изчислителната сложност при генериране на n точки е $O(n(s + \log n))$.

3.1.6 Редица на Фор

Фор създава конструкция на $(0, m, s)$-мрежа и $(0, s)$-редица при основа просто число по-голямо или равно на s, където s е размерността на реди-

цата. Нека $b \geq 2$ е цяло неотрицателно число, със следното представяне $n = n_0 + n_1 b + \ldots + n_m b^m$ при основа b. Дефинираме функцията:

$$\Phi_b(\mathbf{n}) = \frac{n_0}{b} + \frac{n_1}{b^2} + \cdots + \frac{n_m}{b^m},$$

където $\mathbf{n} = (n_0, n_1, \ldots, n_m)^T$. За n-тия елемент от редицата на Фор е в сила следното представяне:

$$x_n = (\Phi_b(P^0\mathbf{n}), \Phi_b(P^1\mathbf{n}), \ldots, (\Phi_b(P^{n-1}\mathbf{n})),$$

където b е просто число по-голямо или равно на s и P^j са степените на матрицата на Паскал P по модул b. Да отбележим, че елементът (u, v) на P е равен на $\binom{v-1}{u-1}$ и че елемента (u, v) на P^j е равен на $j^{v-u}\binom{v-1}{u-1}$.

Може да се покаже, че изчислителната сложност за генериране на една точка на Фор е $O(s(\log_b(n))^2)$.

3.1.7 Точкови множества от тип решетка

Точковите множества тип решетка са семейство от множества точки с нисък дискрепанс традиционно използвани за апроксимация на интеграли от периодични функции с определена степен на гладкост. В последните години се използва следният вид решетка, наречена решетка с ранг 1, с точки във вида:

$$x_n = \left\{\frac{n}{N}g\right\}, \quad n = 0, \ldots, N-1.$$

Тук $g = (g_1, \ldots, g_s)$, наречен генериращ вектор, е s-мерен целочислен вектор, който няма общ множител с N. Скобите около вектора в горната формула означават, че всяка компонента на вектора се замества с дробната си част.

Изборът на добър генериращ вектор, който води до малки грешки при интегриране, не е тривиален. Използват се сложни методи от теория на числата, основани на различни критерии като индекс на Заремба или грешка на най-лошата функция. Корабов ограничава търсенето чрез разглеждане на вектори във вида:

$$g = (1, a, a^2, \ldots, a^{s-1}) \mod N, \quad 1 \leq a \leq N-1, \quad \gcd(a, n) = 1.$$

3.2 Оценки на дискрепанса

Редиците на Собол, Холтън и Фор имат следната форма на оценката на дискрепанс-звезда:

$$D_N^\star \leq C_s \frac{(\log N)^s}{N} + O\left(\frac{(\log N)^{s-1}}{N}\right).$$

Да означим с C_s^H, C_s^F, C_s^S коефициентите в оценките на дискрепанса съответно в редиците на Холтън, Фор и Собол. Тогава

$$C_s^H = \prod_{j=1}^{s} \frac{b_j - 1}{2 \log b_j},$$

където основите b_j за различните размерности са взаимно прости. В работата на Атанасов [8] за дискрепанса на редицата на Холтън е доказана по-добра оценка с константа:

$$C_s^H = \frac{1}{s!} \prod_{j=1}^{s} \frac{b_j - 1}{\ln b_j}.$$

За редицата на Фор съответният коефициент е

$$C_s^F = \frac{1}{s!} \left(\frac{b_s - 1}{2 \log b_s}\right)^s,$$

където b_s е основата на редицата на Фор. Известно е, че коефициентът на Фор притежава свойството $lim_{s\to\infty} C_s^F = 0$. За редицата на Собол

$$C_s^S = \frac{2^{t_s}}{s!(\log 2)^s}.$$

Оценката за t_s може да се намери в [59] и тя е следната:

$$K \frac{s \log s}{\log \log s} \leq t_s \leq \frac{s \log s}{\log 2} + O(s \log \log s),$$

което показва, че t_s расте експоненциално с размерността на редицата s, както и при редицата на Холтън. От друга страна, Фор доказва в [22], че C_s^F е по-малка от C_s^S.

Този факт, обаче, не означава, че редицата на Фор превъзхожда редиците на Холтън и Собол. На практика, асимптотичната оценка на дискрепанса не е много полезна при реални пресмятания. Нещо повече, експерименталните оценки на дискрепанса при практически пресмятания са по-малки от теоретичните.

Освен редиците в оригиналния им вид, има много модификации, наложили се в практиката. Например, известна е модификацията на Холтън, конструирана от Атанасов, [9], за която е доказано, че дава по-добра сходимост от оригиналната редица на Холтън в термините на оценка на дискрепанса.

3.3 Разбъркани редици

С цел подобряване характеристиките и използваемостта на редиците, се извършва модифициране на редиците, известно като разбъркване (scrambling). Подходите за разбъркване се разделят в две основни категории. Едната се основава на рандомизирано изместване (randomized shifting) [7, 16, 41], което има вида:

$$z_n = x_n + r \pmod 1,$$

където x_n е квазислучайно число в интервала $[0,1)^s$, и r е едно s-мерно псевдослучайно число. Използвайки различни псевдослучайни числа, r, получаваме различни разбъркани версии $\{z_n\}$ на оригиналната квазислучайна редица $\{x_n\}$.

Другият подход се основава на цифрови пермутации [39, 54]. Нека $x_n = (x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(s)})$ е едно квазислучайно число в $[0,1)^s$, а $z_n = (z_n^{(1)}, z_n^{(2)}, \ldots, z_n^{(s)})$ е разбъркана (scrambled) версия на x_n. Да предположим, че всяко $x_n^{(j)}$ се представя в система при основа b като $x_n^{(j)} = 0.x_{n_1}^{(j)} x_{n_2}^{(j)} \ldots x_{n_K}^{(j)} \ldots$ и K е броят на цифрите, които ще се разместват. Дефинираме

$$z_n^{(j)} = \sigma(x_n^{(j)}), \quad \text{за} \quad j = 1, 2, \ldots, s,$$

където $z_{n_i}^{(j)} = \phi_i(x_{n_i}^{(j)})$, за $i = 1, 2, \ldots, K$, $\sigma = \{\phi_1, \phi_2, \ldots, \phi_K\}$, и всяко ϕ_i е пермутация на целите числа $\{0, \ldots, b-1\}$. Има различни варианти на метода за разбъркване чрез цифрова пермутация, които се основават на различните дефиниции на ϕ_i, например предложените от Owen [52, 53, 54], Tezuka [63, 64], и Matousek.

3.3.1 Разбъркване на редицата на Холтън

За съжаление, координатите на точки от редицата на Холтън, за която основите b_j са големи прости числа, показват корелации. Този проблем може да се преодолее с леко разместване/разбъркване на точките от редицата по начин, който съхранява ниския дискрепанс на редицата. За първи път този процес формално е описан от Braaten и Weller [10], които дефинират разбъркана обратна функция $S_b(n)$ по аналогия с (3.1.2) като

$$S_b(n) := \frac{\pi_b(d_0)}{b} + \frac{\pi_b(d_1)}{b^2} + \ldots + \frac{\pi_b(D_j)}{b^{j+1}}. \tag{3.3.1}$$

Тук π_b е пермутация на цифрите $(0, 1, \ldots, b-1)$, която оставя фиксирана цифрата 0. Има $(b-1)!$ възможни пермутации. Разбърканата редица на Холтън се дефинира чрез:

$$x_n = (S_{b_1}(n), \ldots, S_{b_s}(n)), \ n = 0, 1, \ldots \tag{3.3.2}$$

Да отбележим, че разбъркването, дефинирано чрез (3.3.1) и (3.3.2) е *детерминистично* разбъркване: всеки набор от пермутации (една пермутация за всяка размерност) винаги води до същата разбъркана версия на редицата на Холтън. Този вид разбъркване е въведен, за да подобри проекциите в ниските размерности на редицата на Холтън.

Редиците с малък дискрепанс намаляват грешката при численото интегриране, но определянето на практически удобна оценка на грешката изобщо не е тривиално, поради твърде грубите граници на грешката, определени от неравенството на Коксма-Хлавка, които са и практически неизползваеми. Действително, практиката в Монте Карло методите да се използва определен критерий за грешка като детерминистично условие за край, е почти невъзможно да се приложи при квази-Монте Карло методите без използване на допълнителна технология. За да се осигури динамична оценка на грешката, и в същото време да се използват квазислучайни редици, се прилагат рандомизирани квази-Монте Карло методи, при които случайността се постига чрез разбъркване на квазислучайните редици или други рандомизиращи техники. Може да се докаже, че при сравнително слаби условия всяко от рандомизираните квази-Монте Карло правила е статистически независимо и може да се използва за формиране на традиционната Монте Карло оценка на грешката, като се използват доверителни интервали, основани на дисперсията на извадката. Най-важният елемент за рандомизираните квази-Монте Карло методи е бърз и ефективен алгоритъм за рандомизиране на квазислучайните редици.

Рандомизираните редици на Холтън са първите редици, които се използват за изясняване на процеса на получаване на практическа оценка на грешката. Това се постига чрез различно рандомизирани редици на Холтън, които след това се използват за оценка на грешката. Различни методи за рандомизиране на редицата на Холтън могат да се намерят в работите на Cranley and Patterson [17], Wang and Hickernell [68] and Morokoff and Caflish [47]. Важно е да се отбележи, че единично рандомизираната редица на Холтън, получена по метода на Кранли-Патерсън и метода за случаен старт на Уанг и Хикернел имат същата корелация на двумерните проекции, както и оригиналната редица. Всъщност, методът на Кранли-Патерсон води до отместване на точките на едно и също разстояние. Използването на различна стойност на n_{start} за всяка размерност също не променя корелацията.

Освен това, за много рандомизиращи методи не е възможно да се напише вида на действителните пермутации π_b, поради което съответните рандомизирани версии на редицата на Холтън не могат да се изразят чрез формулата 3.3.1. Ако по някакъв критерий се търси най-добрия възможен тип разбъркване, то очевидно пространството на търсенето трябва да включва само пермутации, които ясно дефинират стойности за $\pi_b(d_0)$, $\pi_b(d_1), \ldots \pi_b(d_j)$ в 3.3.1.

Методи за разбъркване на редицата на Холтън

Тук ще разгледаме хронологически известните методи за *детерминистично* разбъркване.

Метод на Уорнок

Уорнок [69] използва сегментирана обратна функция:

$$\Psi_b(n) := \frac{(d_0+0) \mod b}{b} + \frac{(d_1+1) \mod b}{b^2} + \ldots + \frac{(d_j+j) \mod b}{b^{j+1}} \quad (3.3.3)$$

вместо (3.3.1), за да дефинира разбъркана версия на редицата на Холтън. Например, $\Psi_3(11) = 0.210\overline{012}_3 = 551/702$, където горната черта означава безкрайно повтаряща се група цифри. Да отбележим, че това разбъркване не е точно от вида (3.3.1), тъй като пермутацията за всяка цифра е различна. Версията на Уорнок на редицата на Холтън изглежда по следния начин:

$$x_n = (\Psi_{b_1}(n), \ldots, \Psi_{b_s}(n)), \; n = 0, 1, \ldots$$

Пермутации на Братен и Уелър

Братен и Уелър [10] използват един алгоритъм (изложен по-долу), за да дефинират π_b от разбърканата обратна функция (3.3.1). Те са табулирали техните пермутации до първите 16 прости числа, при което колкото е по-висока размерността на редицата, толкова е по-добра тяхната версия на редицата (в сравнение с оригиналната редица). Те, обаче, нямат коментар какво се случва при размерност по-голяма от 16.

Алгоритъм на Братен и Уелър: Търсене на пермутации

$\pi_b(0) = 0$

$ChoiceSet = \{1, \ldots, b-1\}$

$for\ i = 1\ to\ b-1\ do$

Избор на $\pi_b(i)$ от $ChoiceSet$ така че да минимизира едномерния дискрепанс на точките $\left(\frac{\pi_b(1)}{b}, \ldots, \frac{\pi_b(i)}{b}\right)$.

$ChoiceSet = ChoiceSet \backslash \{\pi_b(i)\}$.

end do

По-късно, през 2008 Барт Вандевоестин разширява търсенето и публикува таблица на пермутациите до размерност 64. Трябва да се отбележи, че методът на Братен и Уелър изисква значителни изчислителни усилия за определяне на пермутациите поради пресмятането на много едномерни дискрепанси. Например, за пресмятане на пермутация за просто число b, максималният брой умножения M и събирания A е

$$\sum_{k=2}^{b-1} k\left([(b-k)^2]\max + [(b-k)+4] + [(b-k)^2 + 4(b-k) - 1]A\right).$$

Разбира се, това не е проблем, ако размерността s на задачата за интегриране е известна предварително. В този случай е възможно пермутациите да се пресметнат еднократно и да се съхранят за по-късна употреба.

Модификация на Атанасов на редицата на Холтън

За да конструира неговата т.нар. *модифицирана редица на Холтън* Атанасов [8] използва s различни прости числа $b_1, \ldots, b_s$ за основи и избира $\pi_{b_i} = d_j k_i^j \mod b_i$. Целите числа k_i са степени на примитивния корен на $g_i \mod b_i$. Те трябва да бъдат „допустими“ към простите числа b_i, което означава, че те трябва да изпълняват определено условие, за да бъдат използваеми. Метод за конструиране на допустими цели числа k_i е описан в [8], а самите числа могат да бъдат намерени на

`http://parallel.bas.bg/ emanouil/sequences.html`
във файла `halton.dat`. В работата [9] авторите предлагат да се умножава с k_i^{j+1} вместо с k_i^j.

Оптимално разбъркване на Маскани и Чи
Маскани и Чи разглеждат линейно разбъркване, [41]

$$\pi_{b_i}(d_j) = \omega_i d_j \pmod{b_i},$$

при ω_i цяло число. Използвайки критерий от теория на числата, те търсят оптимални ω_i и емпирично проверяват резултатите чрез апроксимация на един определен интеграл. За първите 40 размерности съответните ω_i са табулирани в [41].

3.3.2 Разбъркване на редицата на Фор

Да припомним, че n-тата точка на редицата на Фор се изразява като

$$x_n = (\phi_b(P^0\mathbf{n}), \phi_b(P^1\mathbf{n}), \ldots, \phi_b(P^{s-1}\mathbf{n})),$$

където матрично-векторните произведения $P^j n$ за $j = 0, \ldots, s-1$ се пресмятат по модул b, при b просто число по-голямо или равно на размерността s, а P е матрицата на Паскал по модул b, чийто елемент (i, j) е равен на $\binom{j-1}{i-1} \mod b$. Както при редицата на Холтън, тази конструкция води до корелация между отделните координати, което се изразява в корелирани точки в проекциите на редицата в ниските размерности. Решението на този проблем се състои в случайно разбъркване на редицата на Фор.

Схема на Оуен за разбъркване

Оуен предлага схема за рандомизиране на произволни квазислучайни редици и точкови мнжества, основани на цифрови пермутации. Нека $x_n = (x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(s)}) \in [0,1)^s$ и $z_n = (z_n^{(1)}, z_n^{(2)}, \ldots, z_n^{(s)})$ е разбърканата версия на x_n. Предполагаме, че всяко $x_n^{(i)}$ може да се представи при основа b като $x_n^{(i)} = \sum_{j=1}^{+\infty} x_{n,j}^{(i)} b^{-j}$, след което за рандомизираната точка $z_n^{(i)}$ полагаме $z_n^{(i)} = \sum_{j=1}^{+\infty} z_{n,j}^{(i)} b^{-j}$ със $Z_{n,j}^{(i)}$ дефинирани като случайни пермутации на $x_{n,j}^{(i)}$:

$$z_{n,1}^{(i)} = \pi_i(x_{n,1}^{(i)})$$

$$z_{n,2}^{(i)} = \pi_{i,x_{n,1}^{(i)}}(x_{n,2}^{(i)})$$

$$z_{n,3}^{(i)} = \pi_{i,x_{n,1}^{(i)},x_{n,2}^{(i)}}(x_{n,3}^{(i)})$$

...

$$z_{n,k}^{(i)} = \pi_{i,x_{n,1}^{(i)},x_{n,2}^{(i)},\ldots,x_{n,k-1}^{(i)}}(x_{n,k}^{(i)}).$$

Всяка пермутация π е равномерно разпределена измежду $b!$ пермутации на $\{0,1,\ldots,b-1\}$ и пермутациите са взаимно независими. Първата цифра в представянето на $x_n^{(i)}$ при основа b се пермутира чрез π_i за всяко n. Втората цифра се пермутира чрез пермутацията $\pi_{i,x_{n,1}^{(i)}}$, която зависи от стойността на първата цифра $x_{n,1}^{(i)}$. Аналогично, пермутацията, приложена към k-тата цифра $x_{n,k}^{(i)}$ зависи от стойностите на първите $k-1$ цифри $x_{n,1}^{(i)}$ до $x_{n,k-1}^{(i)}$.

Този метод за разбъркване няма лесна и ефективна практическа реализация, поради необходимостта от много проверки и памет. Известни са много опростени версии на метода на Оуен за разбъркване. Най-широко разпространени са методите за линейно разбъркване. Те са лесни за изпълнение и позволяват бърза генерация на разбъркани редици. Формулите за линейно разбъркване са подобни на рекуренциите, използвани в линейните конгруентни генератори на псевдослучайните числа.

Обобщена редица на Фор

Обобщената редица на Фор, известна като GFaure в литературата, е предложена от Тезука, като за размерност j генераторната матрица има вида $C^{(j)} = A^{(j)}P^{j-1}$, където $A^{(j)}$ за $j = 1,\ldots,s$ са произволни несингулярни долни триъгълни матрици. Програмният код на езика C за редицата GFaure може да се намери в [62].

Специален случай на GFaure е разгледан в [24], при който редиците използват за $A^{(j)}$ долна триъгълна матрица с елементи, равни на 1 за всяко j. Ако $A^{(j)}$ е само диагонална, тогава тази редица ще бъде оригиналната редица на Фор.

Случайно линейно (цифрово) разбъркване

Няколко опростени версии на разбъркването на Оуен, в които се използват матрици и изместващи вектори, са дискутирани от Матушек в [42]. Точките от редица, конструирана по методите на Матушек, има формата

$$x_n = \left(\phi_b(A^{(1)}P^0 n + g_1), \phi_b(A^{(2)}P^1 n + g_2), \ldots, \phi_b(A^{(s)}P^{s-1} n + g_s)\right).$$

Матриците $A(1), \ldots, A(s)$ са различни случайно разбъркващи матрици от определен вид. Векторите $g_1, \ldots, g_s$ са различни вектори за случайно отместване. За случайно линейно разбъркване имаме

3.3.3 Разбъркване на редицата на Собол

Общите методи за разбъркване (например, методът на Оуен) са приложими за редицата на Собол, но има някои ограничения. Например, линейната пермутация не е подходяща за редицата на Собол по следните причини:

Нека $x_n = (x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(s)})$ е квазислучайно число в $[0,1)^s$ и $z_n = (z_n^{(1)}, z_n^{(2)}, \ldots, z_n^{(s)})$ е разбърканата версия на точката x_n. Да предположим, че всяко $x_n^{(j)}$ има представяне при основа b като $x_n^{(j)} = 0.x_{n1}^{(j)} s_{n2}^{(j)} \ldots x_{nK}^{(j)} \ldots$, където K дефинира броя на цифрите за разбъркване във всяка точка. Дефинираме

$$z_n^{(j)} = c_1 x_n^{(j)} + c_2, \;\; j = 1, 2, \ldots, s,$$

където $c_1 \in \{1, 2, \ldots, b-1\}$ и $c_2 \in \{1, 2, \ldots, b-1\}$. Тъй като редицата на Собол е построена над F_2, то трябва да присвоим 1 на c_1 и 0 или 1 на c_2. Тъй като изборът на c_1 е критично важен за качеството на разбърканата редица на Собол, този метод за линейно разбъркване не е подходящ за редицата на Собол или коя да е друга редица над F_2.

Да припомним, че качеството на редицата на Собол силно зависи от избора на направляващите вектори. Корелациите между различните размерности са породени от неподходящ избор на направляващите вектори. Има методи, които подобряват редицата на Собол чрез внасяне на повече равномерност в началните направляващи вектори, но няма адекватен начин за проверка на това подобрение. Тук ще разгледаме един начин за разбъркване на редицата на Собол независим от началните направляващи вектори. Методът е предложен от Маскани и Чи.

Идеята на този метод е да се направи разбъркване на k бита от редицата на Собол вместо на една цифра в даден момент. Нека x_n е n-тата точка на Собол и k бита от x_n ще бъдат разбъркани. Нека z_n е разбърканата версия на x_n. Процедурата е следната:

1. $y_n = \lfloor x_n * 2^k \rfloor$, са k най-значещи бита на x_n.
2. $y_n^* = a y_n \pmod m$ и $m \geq 2^k - 1$, е линейното разбъркване, приложено към това цяло число.
3. $z_n = \frac{y_n^*}{2^k} + (x_n - \frac{y_n}{2^k})$, е обратното вмъкване на разбърканите битове в точката на Собол.

Ключовият момент в този подход е използването на подходящ линеен конгруентен генератор. Тези генератори се използват с модул просто число или степен на две. Когато модулът е степен на две, изпълнението е изчислително евтино и бързо. Недостатък е, че трудно се постига желаното качество на числата. По-добро качество се постига с модул просто число. Процедурата е тествана числено с прости числа на Мерсен и Софи-Жермен.

Глава 4

Монте Карло методи

В тази глава ще разгледаме Монте Карло методите за приближено пресмятане на интеграли. Те се основават на факта, че интеграл от функция $f(x)$, интегуема по Лебег, може да се разглежда като математическото очакване на функцията f, оценена в случайна точка от областта, в която интегрираме.

Разглеждаме интеграл върху едномерния единичен интервал:

$$I[f] = \int_0^1 f(x)dx = \overline{f}. \tag{4.0.1}$$

Нека x е случайна величина, равномерно разпределена в единичния интервал. Тогава

$$I[f] = E[f(x)]. \tag{4.0.2}$$

За интеграл върху единичния куб $I^d = [0,1]^d$ с размерност d,

$$I[f] = E[f(x)] = \int_{I^d} f(x)dx, \tag{4.0.3}$$

където x е равномерно разпределен вектор в единичния куб.

Квадратурната формула от тип Монте Карло се основава на вероятностната интерпретация на интеграла. Разглеждаме редица $\{x_n\}$, извадка от стандартното равномерно разпределение в интервала $[0,1]$, и следната емпирична апроксимация на математическото очакване:

$$I_N[f] = \frac{1}{N}\sum_{n=1}^{N} f(x_n). \tag{4.0.4}$$

Съгласно усиления закон за големите числа (Feller 1971), тази апроксимация е сходяща с вероятност едно, т.е.,

$$I_N[f] \to I[f]. \tag{4.0.5}$$

При това, оценката е неизместена, което означава, че математическото очакване на $I_N[f]$ е точно $I[f]$ за всяко N, т.е.,

$$E[I_N[f]] = I[f], \tag{4.0.6}$$

където усредняването е върху избраните точки $\{x_n\}$.

Дефинираме грешката на интегриране по метод Монте Карло като

$$\epsilon_N[f] = I[f] - I_N[f] \tag{4.0.7}$$

така че отклонението е $E[\epsilon_N[f]]$ и средно-квадратичната грешка е

$$E[\epsilon_N[f]^2]^{1/2}. \tag{4.0.8}$$

4.1 Точност на обикновения Монте Карло метод

Грешката при приближеното пресмятане на интеграли по метод Монте Карло се определя от Централната гранична теорема (Feller 1971).

Теорема 4.1.1 *За големи N*

$$\epsilon_N[f] \approx \sigma N^{-1/2} \nu \tag{4.1.1}$$

където ν е стандартна нормална ($N(0,1)$) случайна величина и константата $\sigma = \sigma[f]$ е квадратен корен от дисперсията на f, т.е.,

$$\sigma[f] = \left(\int_{I^d} (f(x) - I[f])^2 \right)^{1/2}. \tag{4.1.2}$$

Следното твърдение е по-прецизно:

$$\lim_{N\to\infty} \Pr(a < \frac{\sqrt{N}}{\sigma} \epsilon_N < b) = \Pr(a < \nu < b) = \int_a^b (2\pi)^{-1/2} e^{-x^2/2} dx. \tag{4.1.3}$$

Това твърдение показва, че грешката при Монте Карло интегрирането има порядък $O(N^{-1/2})$ с константа, която е точно корен квадратен от дисперсията на случайната величина $f(x)$. Нещо повече, статистическото разпределение на грешката е приблизително нормална случайна величина. За

разлика от обичайните резултати в числените методи, това е вероятностен резултат. Той не дава абсолютна горна граница на грешката; вместо това показва, че грешката има определен порядък с определена вероятност. От друга страна, този резултат е равенство, така че границите, които той определя са тесни.

Сега ще дадем частично доказателство на Централната гранична теорема, което доказва, че порядъкът на грешката е $O(N^{-1/2})$. Извеждането на Гаусовото разпределение на грешката е по-трудно (Feller 1971). Първо дефинираме $\xi_i = \sigma^{-1}(f(x_i) - \overline{f})$, където x_i е равномерно разпределена случайна величина. Тогава

$$E[\xi_i] = 0,$$

$$E[\xi_i^2] = \int_{I^d} \sigma^{-2}(f(x_i) - \overline{f})^2 dx = 1.$$

$$E[\xi_i \xi_j] = 0 \ \ if \ \ i \neq j. \tag{4.1.4}$$

Последното равенство е вярно поради независимостта на x_i, $i = 1, \ldots, N$. Сега да разгледаме сумата

$$S_N = N^{-1} \sum_{i=1}^{N} \xi_i = \sigma^{-1} \epsilon_N. \tag{4.1.5}$$

Нейната дисперсия е:

$$E[S_N^2]^{1/2} = E\left[N^{-2}\left(\sum_{i=1}^{N} \xi_i\right)^2\right]^{1/2}$$

$$= N^{-1} E\left[\sum_{i=1}^{N} \xi_i^2\right] + E\left[\sum_{i=1}^{N} \sum_{j \neq i} \xi_i \xi_j\right]^{1/2}$$

$$= N^{-1}\left\{\sum_{i=1}^{N} 1 + 0\right\}^{1/2}$$

$$= N^{-1/2}.$$

Следователно,

$$E[\epsilon_N^2] = \sigma N^{-1/2}, \tag{4.1.6}$$

което показва, че грешката има порядък $O(\sigma N^{-1/2})$.

Обратното твърдение на Централната гранична теорема е удобно за определяне на размера N на извадката. Тъй като границата на грешката в ЦГТ е вероятностна, точността на Монте Карло метода за интегриране може да се осигури само за определено доверително ниво. За да се гарантира грешка по-малка от дадено число ε при ниво на доверие c, е необходимо извадката да има N точки, където

$$N = \varepsilon^{-2}\sigma^2 s(c), \tag{4.1.7}$$

където s е функцията на доверие за нормална променлива, т.е.,

$$c = \int_{-s(c)}^{s(c)} e^{-x^2/2} dx/\sqrt{2\pi} = \mathrm{erf}(s(c)/\sqrt{2}).$$

Например, 95 процента доверие в размера на грешката изисква приблизително $s = 2$.

За дадено приложение, точната стойност на дисперсията е неизвестна (трудно е пресмятането на самия интеграл), така че формулата (4.1.7) не може да се използва директно. Има лесен начин за преодоляване на това чрез определяне на емпиричната грешка и дисперсия (Hogg and Craig 1995). Извършваме M пресмятания използвайки независими точки x_i за $1 \leq i \leq MN$. За всяко j получаваме стойности $I_N^{(j)}$ за $1 \leq j \leq M$. Емпиричната грешка е $\bar{\epsilon}_N$:

$$\bar{\epsilon}_N = \left(M^{-1} \sum_{j=1}^{M} (I_N^{(j)} - \bar{I}_N)^2 \right)^{1/2}, \tag{4.1.8}$$

където

$$\bar{I}_N = M^{-1} \sum_{j=1}^{M} I_N^{(j)}. \tag{4.1.9}$$

Емпиричното стандартно отклонение $\bar{\sigma}$ е:

$$\bar{\sigma} = N^{1/2} \bar{\epsilon}_N. \tag{4.1.10}$$

Тази стойност може да се използва за σ в (4.1.7), за да се определи N при дадена точност ϵ и дадено ниво на доверие c.

4.2 Сравнение с мрежовите методи

Да сравним скоростта на сходимост на метод Монте Карло с тази на д
терминистичен мрежов метод за приближено интегриране, например пр
вилото на Симпсон. Скоростта на сходимост на мрежова квадратура $O(N^{-k/d})$ за метод от ред k и размерност d, тъй като мрежа с N точки в единичния куб има стъпка $N^{-1/d}$. От друга страна, скоростта на сходимост на Монте Карло е $O(N^{-1/2})$ независимо от размерността. Така че Монте Карло методът е по-добър (по отношение на скоростта на сходимост) от мрежов метод при висока размерност, ако

$$k/d < 1/2.$$

От друга страна, за аналитична функция в периодична област стойността на k е безкрайна, така че това просто обяснение пропада. Едно по-реалистично обяснение на робастността на Монте Карло е това, че практически е невъзможно да се построи мрежа при голяма размерност. Най-простата кубична мрежа в размерност d изисква 2^d точки. За $d = 20$, което не е особено голяма размерност, това са повече от милион точки. Нещо повече, практически невъзможно е да се направи рафиниране на мрежа при висока размерност, тъй като рафинирането изисква нарастване на броя на точките с множител 2^d. В противовес на тези трудности за мрежа при висока размерност, точността на квадратурна формула Монте Карло е почти независима от размерността и всяка допълнителна точка, добавена в Монте Карло квадратурата, дава последващо подобряване на точността. Всъщност, стойността на N, при която оценката на грешката $O(N^{-1/2})$ става вярна, е трудно да бъде предсказана, но опитът показва, че за средна размерност (например, $d = 20$), грешка с размер $O(N^{-1/2})$ се достига за разумни стойности на N.

Две други интерпретации на Монте Карло квадратурите са смислени и полезни. Да разгледаме реда на Фурие за периодична функция с период едно

$$f(x) = \sum_{k=-\infty}^{\infty} \widetilde{f}(k) e^{2\pi i k x}.$$

Интегралът $I[f]$ е точно $\widetilde{f}(0)$, т.е., приносите към интеграла са 0 от всички вълнови числа $k \neq 0$. За мрежа със стъпка $1/n$, мрежовата квадратурна формула е

$$I_n^{(g)} = n^{-1} \sum_{i=1}^{n} f(i/n).$$

Приносите към тази сума са 0 (както трябва да бъде) от вълнови числа $k \neq mn$ за цяло число m, и $\widetilde{f}(k)$ за вълнови числа $k = mn$. Следователно точността на мрежовата квадратура е 100 % при $k \neq mn$, но 0 % при $k = mn$ (и $m \neq 0$). Монте Карло квадратурата използва случаен масив и е частично точна за всички k, което е по-добър резултат от този на мрежовия метод, ако коефициентите на Фурие намаляват бавно.

Накрая, при сравнение на изпълнението на мрежовите и Монте Карло методите трябва да се вземе пред вид и генерирането на точките при висока размерност. За регулярна мрежа, преходът от една към друга точка става с промяна на една координата във всеки определен момент. За много задачи, това е неефективен начин за използване на непроменените координати. От друга страна, в случаен масив всички компоненти се променят за всяка точка и пространството се покрива по-пълно. Това е в съгласие с общата природа на Монте Карло - всяка точка в Монте Карло формулата за интегриране е оценка на интеграла върху цялата област.

4.3 Намаляване на дисперсията

При приближено интегриране по метод Монте Карло грешката ϵ и размера на извадката N са свързани чрез:

$$\epsilon = O(\sigma N^{-1/2}),$$

$$N = O(\sigma/\epsilon)^2.$$

Изчислителното време, което методът изисква, е пропорционално на броя на реализациите N и следователно има порядък $O(\sigma/\epsilon)^2$, т.е., изчислителното време нараства много бързо, ако се изисква по-добра точност. Има две възможности за ускоряване на сходимостта (намаляване на грешката) на Монте Карло методите. Първата е намаляване на дисперсията, при която подинтегралната функция се трансформира по такъв начин, че да се намали дисперсията σ^2. Втората възможност е да се модифицира използваната случайна редица, т.е., да се замени чрез алтернативна редица, която подобрява степенния показател $-1/2$ във втория множител на израза за грешката. Основните методи за намаляване на дисперсията са *симетризация на подинтегралната функция; групиране на извадките; метод на съществената извадка; интегриране по части.*

4.3.1 Симетризация на подинтегралната функция

Този метод работи по следния начин: освен всяка точка (многомерна) x, участваща в квадратурната формула, използваме също $-x$. В резултат получаваме следното Монте Карло квадратурно правило:

$$I_N[f] = \frac{1}{2N}\sum_{n=1}^{N}(f(x_n) + f(-x_n)). \tag{4.3.1}$$

Мотивацията за този метод е представянето на подинтегралната функция в ред на Тейлор за малки стойности на дисперсията. Да разгледаме очакването на $E[f(x)]$, където x е случайна величина $N(0, \sigma^2)$ и σ е малка. Полагаме

$$x = \sigma\widetilde{x}.$$

Представянето на $f = f(\sigma\widetilde{x})$ в ред на Тейлор за малки σ е:

$$f = f(0) + f'(0)\sigma\widetilde{x} + O(\sigma^2).$$

Тъй като разпределението на $\widetilde{x}$ е симетрично относно 0, очакването $E[\widetilde{x}]$ на линейния член е нула. В обикновения Монте Карло този член не се нулира, така че грешката е пропорционал на σ. При симетризацията, линейния член се нулира, така че грешката е пропорционална на σ^2.

4.3.2 Отделяне на главната част

Идеята на този метод е да се използва функция g, която е близка до f и за която $I[g] = \int_{I^d} g(x)$ е известен. Интегралът $I[f]$ се записва по следния начин:

$$\int_{I^d} f(x)dx = \int_{I^d}(f(x) - g(x))dx + \int_{I^d} g(x)dx.$$

Обикновеният Монте Карло метод се прилага за приближено пресмятане на интеграл от $f - g$ в разглежданата област, при което се получава следната квадратурна формула:

$$I_n[f] = \frac{1}{N}\sum_{n=1}^{N}(f(x_n) - g(x_n)) + I[g].$$

Грешката при интегрирането $\varepsilon_N[f] = I[f] - I_N[f]$ има порядък:

$$\varepsilon_N[f] \approx \sigma_{f-g}N^{-1/2},$$

където дисперсията е

$$\sigma^2_{f-g} = \int_{I^d} (\widetilde{f}(x) - \widetilde{g}(x))^2 dx$$

при използване на означението

$$\widetilde{f}(x) = f(x) - I[f], \;\; \widetilde{g}(x) = g(x) - I[g].$$

Това показва, че методът на отделяне на главната част е ефективен, ако

$$\sigma_{f-g} \ll \sigma_f.$$

За оптималното използване на този метод въвеждаме множител λ. За дадена функция g записваме интеграла от f като

$$\int_{I^d} f(x)dx = \int_{I^d} (f(x) - \lambda g(x))dx + \lambda \int_{I^d} g(x)dx.$$

Грешката в Монте Карло квадратурата за първия интеграл е пропорционална на дисперсията:

$$\sigma^2_{f-\lambda g} = \int_{I^d} (\widetilde{f}(x) - \lambda \widetilde{g}(x))^2 dx.$$

Оптималната стойност на λ се получава при минимизиране на $\sigma^2_{f-\lambda g}$:

$$\lambda = E[\widetilde{f}\widetilde{g}] = \left(\int_{I^d} \widetilde{f}\widetilde{g}dx\right) \Big/ \left(\int_{I^d} \widetilde{g}^2 dx\right).$$

4.3.3 Метод на съответстващите моменти

Грешката при интегрирането по метод Монте Карло частично се поражда от статистическата грешка, свързана с извадката, т.е., разликата между желаната плътност $p(x)$ и емпиричната плътност на точките от извадката $\{x_n\}_{n=1}^N$. Част от тази разлика може да се види директно чрез сравнение на моментите на двете разпределения. Дефинираме първия и втория момент m_1 и m_2 на p като

$$m_1 = \int_{I^d} xp(x)dx, \;\; m_2 = \int_{I^d} x^2 p(x)dx.$$

Първият и вторият моменти μ_1 и μ_2 на извадката $\{x_n\}_{n=1}^N$ са:

$$\mu_1 = N^{-1}\sum_{n=1}^{N} x_n, \quad \mu_2 = N^{-1}\sum_{n=1}^{N} x_n^2.$$

Грешката се поражда от неравенството на тези моменти, т.е.,

$$\mu_1 \neq m_1, \quad \mu_2 \neq m_2.$$

Частично коригиране на статистическата грешка на извадката може да се извърши чрез трансформация, която води до съответствие на моментите. Това може да бъде направено чрез проста трансформация на точките от извадката. За съответствие на първия момент на извадката с този на p, заместваме x_n чрез

$$y_n = (x_n - \mu_1) + m_1.$$

То удовлетворява

$$N^{-1}\sum y_n = m_1$$

така че първият момент е точно същия. За съответствие на първите два момента, заместваме x_n чрез

$$y_n = (x_n - \mu_1)/c + m_1, \quad c = \sqrt{\frac{m_2 - m_1^2}{\mu_2 - mu_1^2}},$$

$$N^{-1}\sum u_n = m_1, \quad N^{-1}\sum u_n^2 = m_2.$$

Тези трансформирани редици с правилни моменти трябва да се използват с повишено внимание. Тъй като трансформацията включва μ_1 и μ_2, точките от новата извадка вече не са независими, поради което Монте Карло грешката не е вече дефинираната по-рано. Например, ЦГТ не е директно приложима и оценката може да е изместена. В действителност, това е пример на втория подход за ускоряване на сходимостта на Монте Карло метода – посредством модификация на свойствата на случайната редица. Той е представен тук заедно с методите за намаляване на дисперсията, защото подобрението в сходимостта, което този подход дава, е сравнимо с подобрението, получено чрез намаляване на дисперсията.

Pullin (1979) е формулирал метод, в който точките от извадката са независими, и имат предписаното емирично очакване и дисперсия, но произлизат от гаусово разпределение със случайно избрани очакване и дисперсия.

4.3.4 Стратификация

Стратификацията комбинира предимствата от използването на регулярна мрежа (грид) и предимствата от използването на случайни точки. В най-простия случай, стратификацията се основава на регулярен грид и едномерна равномерна плътност. Разделяме областта на интегриране $\Omega = [0,1]$ на M части Ω_k дефинирани чрез:

$$\Omega_k = \left[\frac{k-1}{M}, \frac{k}{M}\right].$$

Така, всяко подмножество има една и съща мярка, $|\Omega_k| = 1/M$. Дефинираме средните върху всяко Ω_k чрез

$$\overline{f}(x) = \overline{f}_k = |\Omega_k|^{-1} \int_{\Omega_k} f(x)dx \quad for \ \ x \in \Omega_k.$$

За всяко k, генерираме $N_k = N/M$ точки $x_i^{(k)}$, равномерно разпределени в Ω_k. Тогава, стратифицираната квадратура е просто сумата от квадратурите върху всяко подмновество, а именно,

$$I_N = N^{-1} \sum_{k=1}^{M} \sum_{i=1}^{N/M} f(x_i^{(k)}.$$

Монте Карло квадратурната грешка за тази стратифицирана сума е:

$$\varepsilon \approx N^{-1/2}\sigma_s,$$

$$\sigma_s^2 = \int_{\Omega} (f(x) - \overline{f}(x))^2 dx = \sum_{k=1}^{M} \int_{\Omega_k} (f(x) - \overline{f}_k)^2 dx.$$

За това стратифицирано квадратурно правило има прост резултат. Стратифицираната Монте Карло квадратура винаги е по-добра от нестратифицираната, защото:

$$\sigma_s \leq \sigma.$$

Доказателството на това неравенство е директно. За всяко k, $c = \overline{f}_k$ минимизира интеграла:

$$\int_{\Omega_k} (f(x) - c)^2 dx.$$

В частност,

$$\int_{\Omega_k}(f(x)-\overline{f}_k)^2dx \le \int_{\Omega_k}(f(x)-\overline{f})^2dx.$$

Сумираме по k и получаваме

$$\begin{aligned}\sigma_s &= \sum_{k=1}^{M}\int_{\Omega_k}(f(x)-\overline{f}_k)^2dx \\ &\le \sum_{k=1}^{M}\int_{\Omega_k}(f(x)-\overline{f})^2dx \\ &= \sigma^2.\end{aligned}$$

Стратификацията може да се обобщи по следния начин. Разделяме областта Ω на M части Ω_k със

$$\Omega = \bigcup_{k=1}^{M}\Omega_k.$$

Вземаме N_k случайни променливи във всяка част Ω_k при

$$\sum_{k=1}^{M}N_k = N.$$

Във всяко подмножество Ω_k избираме точки $x_n^{(k)}$, разпределени с плътност $p^{(k)}(x)$, която

$$\begin{aligned}p^{(k)}(x) &= p(x)/\overline{p}_k, \\ \overline{p}_k &= \int_{\Omega_k}p(x)dx.\end{aligned}$$

Стратифицираната квадратурна формула е следната сума по k:

$$I_N[f] = \sum_{k=1}^{M}\frac{\overline{f}_k}{N_k}\sum_{n=1}^{N_k}f(x_n^{(k)}). \tag{4.3.2}$$

Грешката при интегрирането е:

$$\epsilon_N[f] = I[f]-I_N[f] = \sum_{k=1}^{M}\epsilon_{N_k}^{(k)}[f].$$

Компонентите на тази грешка са:

$$\epsilon_{N_k}^{(k)}[f] \approx N_k^{-1/2}\overline{p}_k \left(\int_{\Omega_k} (f(x) - \overline{f}_k)^2 p^{(k)}(x)dx \right)^{1/2}$$

$$= (\overline{p}_k/N_k)^{1/2}\sigma^{(k)},$$

в които дисперсиите са:

$$\sigma^{(k)} = (\overline{p}_k)^{1/2} \left(\int_{\Omega_k} (f(x) - \overline{f}_k)^2 p^{(k)}(x)dx \right)^{1/2}$$

$$= \left(\int_{\Omega_k} (f(x) - \overline{f}_k)^2 p(x)dx \right)^{1/2},$$

и очакванията са:

$$\overline{f}_k = \int_{\Omega_k} f(x)p(x)dx/\overline{p}_k.$$

Стратификацията винаги намалява грешката при интегриране, ако разпределението на точките е балансирано. Условието за балансираност е, че за всички k да е изпълнено:

$$\overline{f}_k/N_k = 1/N,$$

т.е., броят на точките в множеството Ω_k е пропорционален на неговия претеглен размер $\overline{p}_k$. Грешката на стратифицираната квадратура е:

$$\epsilon_N \approx N^{-1/2}\sigma_s, \;\; \sigma_s^2 = \sum_{k=1}^{M} \sigma^{(k)2}.$$

Тъй като дисперсията върху подмножество винаги е по-малка от дисперсията върху цялото множество, т.е., $\sigma_s \leq \sigma$, стратификацията винаги понижава грешката. В действителност, по-добър избор от условието за балансираност може да бъде да се генерират повече точки, където f има голяма дисперсия.

4.3.5 Метод на съществената извадка (Importance sampling)

Този метод е най-често използвания Монте Карло метод с намаляване на дисперсията. Разглеждаме интеграла $\int f(x)$ и, въвеждайки плътност p, го записваме в следния вид:

$$I[f] = \int f(x)dx = \int \frac{f(x)}{p(x)}p(x)dx.$$

Сега да разгледаме този интеграл като интеграл от случайната величина $f(x)/p(x)$ с плътност $p(x)$. Генерираме точки x_n с плътност $p(x)$ и формираме Монте Карло оценката

$$I_N[f] = \frac{1}{N}\sum_{n=1}^{N}\frac{f(x_n)}{p(x_n)}.$$

Резултантната грешка $\epsilon_N[f] = I[f] - I_N[f]$ има порядък

$$\epsilon_N[f] \approx \delta_p N^{-1/2},$$

в която дисперсията δ_p е:

$$\delta_p = \int \left(\frac{f(x)}{p(x)} - I\right)^2 p(x)dx.$$

По този начин грешката на Монте Карло квадратурата се намалява, ако

$$\delta_p \ll \delta.$$

Методът на съществената извадка е ефективен, когато f/p е приблизително константа, така че δ_p е малко число. Трудността при този метод е, че се изисква генериране на извадка от случайна величина с плътност на разпределение p, но за това може да се приложи метода на селекцията, ако е необходимо.

Методът на съществената извадка, първоначално предложен от Кан [31], вероятно е най-широко използваният Монте Карло метод с намалена дисперсия. При използването на този метод се акцентира върху редки, но важни събития, т.е., малки подобласти, в които абсолютната стойност на подинтегралната функция е голяма. Една от трудностите в този метод е, че се изисква моделиране на случайна величина с плътност, пропорционална на подинтегралната функция, което не винаги е възможно – в такъв случай

се използва метода на селекцията. Известно е също така, че при този метод е възможно рязко увеличаване на дисперсията в някои случаи.

В работата [28] е предложен метод на защитена съществена извадка - когато е комбиниран с подходящи управляващи променливи, този метод винаги има по-добра (или равна) дисперсия с обикновения Монте Карло метод. Грешката на този метод, обаче, може да бъде много по-голяма от грешката на метода на съществената извадка. Оуен и Зу [55] предлагат съществена извадка, при която случайната величина се генерира със сума от m плътности при m управляващи променливи (по една за всеки елемент от сумата). Доказано е, че грешката при този метод никога не е по-лоша от грешката на метода на съществената извадка с плътност кой да е от компонентите на сумата от плътности.

Друг вариант, многомерна съществена извадка, е представен в [66] и [67]. Идеята е, че в някои подобласти подинтегралната функция може да бъде пропорционална на една от плътностите, докато други плътности са подходящи за други подобласти. Целта е да се поставят по-големи тегла на локално най-подходящите плътности.

4.3.6 Метод на разделяне по важност (Importance separation)

В работите [34, 35] е описан метод, наречен разделяне по важност, който комбинира идеята на метода на съществената извадка с разделяне на областта на подобласти. Този метод има най-добра скорост на сходимост за определен клас функции, но негов недостатък е нарастващата изчислителна сложност при увеличаване на размерността на задачата.

Нека $W^{(r)}(M;G)$ е класът от функции $f(x)$, непрекъснати върху G и с непрекъснати r-ти частни производни, такива че $|D^r f(x)| \leq M$, където $D^r = D_1^{r_1} \dots D_d^{r_d}$ е r-тата производна, $r = (r_1, r_2, \dots, r_d)$, $|r| = r_1 + r_2 + \dots + r_d$, а $D_i = \frac{\partial}{\partial x_i}$. Първо разглеждаме едномерната задача за пресмятане на интеграли. Разделяме интервала $[0,1]$ на N подинтервала (предполагаме, че $f(x) \neq 0$ в $[0,1]$):

$$x_0 = 0,\ x_N = 1,\ D_i \equiv [x_{i-1}, x_i],$$

$$x_i = x_{i-1} + \frac{C_i}{f(x_{i-1}(N-i+1)}},\ i = 1, \dots, N-1, \qquad (4.3.3)$$

където

$$C_i = 1/2[f(x_{i-1}) + f(1)](1 - x_{i-1}),\ i = 1, \dots, N-1.$$

Тази схема дефинира едно разделяне по важност на областта $D \equiv [0,1]$.

Изпълнено е:

$$I = \int_0^1 f(x)p(x)dx = \sum_{i=1}^{N} \int_{x_{i-1}}^{x_i} f(x)p(x)dx.$$

Ако $f(x) \in H(1,L)_{[0,1]}$, то съществуват константи L_i, такива че

$$L_i \geq \left|\frac{\partial f}{\partial x}\right| \text{ за всяко } x \in D_i. \tag{4.3.4}$$

При това, за горната схема съществуват константи c_{1_i} c_{2_i}, такива че

$$p_i = \int_{D_i} p(x)dx \leq c_{1_i}/N, \;\; i = 1, \ldots, N \tag{4.3.5}$$

и

$$\sup_{x_{1_i}, x_{2_i} \in D_i} |x_{1_i} - x_{2_i}| \leq c_{2_i}/N, \; i = 1, \ldots, N. \tag{4.3.6}$$

Теорема 4.3.1 *Нека $f(x) \in H(1,L)_{[0,1]}$. Тогава грешката на метода на разделяне по важност на областта D е:*

$$\varepsilon_N \approx \sqrt{2} \left[\frac{1}{N} \sum_{j=1}^{N} (L_j c_{1_j} c_{2_i})^2\right]^{1/2} N^{-3/2}. \tag{4.3.7}$$

Сега да разгледаме случая на многомерни интеграли:

$$I[f] = \int_D f(x)p(x)dx, \;\; x \in D \subset R^d,$$

където $f \in W^{(1)}(L; D)$.

Нека $f(x) \in W^{(1)}(L_i; d_i)$ за всяко $x \in d_i = [x_{i-1}, x_i]$. Предполагаме, че има разделяне по важност на областта по аналогия с едномерния случай (на подобласти, които са равномерно малки по обем и по вероятност). Нека съществуват вектори $L_i = (L_{i_1}, \ldots, L_{i_d})$, $i = 1, \ldots, N$, такива че

$$L_{i_l} \geq \left|\frac{\partial f}{\partial x_i}\right|, \; l = 1, \ldots, d, \;\; x = (x_1, \ldots, x_d) \in d_i.$$

Освен това, нека съществуват константи c_{1_i} и вектори $c_{2_i} = (c_{2_i 1}, c_{2_i 2}, \ldots, c_{2_i d})$, такива че

$$p_i = \int_{d_i} p(x)dx \leq c_{1_i}/N,\ i = 1, \ldots, N$$

и

$$d_{il} = \sup_{x_{i_{l1}}, x_{i_{l2}} \in d_i} |x_{i_{l1}} - x_{i_{l2}}| \leq c_{2_i l}/N^{1/d},\ i = 1, \ldots, N,\ l = 1, \ldots, d.$$

В многомерния случай аналогично е в сила следното твърдение:

$$\varepsilon_N \approx \sqrt{2}d \left[\frac{1}{N} \sum_{i=1}^{N} (L_i c_{1_i} c_{2_i})^2 \right]^{1/2} N^{-1/2-1/d}.$$

Недостатък на гореописания метод е повишаването на изчислителната сложност. Точността е подобрена (в действителност, метода дава теоретично оптималната точност), но за сметка на увеличения брой на допълнителните пресмятания, което прави този метод практически неефективен при голяма размерност.

4.3.7 Руска рулетка

Някои Монте Карло пресмятания включват безкрайни суми, например при итериране на интегрални уравнения. Методът на руската рулетка превръща безкрайна сума в сума с крайна дължина за всяка извадка. Разглеждаме сумата

$$S = \sum_{n=0}^{\infty} a_n$$

и предполагаме, че членовете a_n са експоненциално намаляващи, т.е.,

$$|a_n| \leq c\lambda^n,$$

където $0 < \lambda < 1$. Избираме $\lambda < k < 1$, и нека M да бъде избрано в съответствие с дискретното експоненциално разпределение, така че:

$$\Pr(M \geq n) = k^n.$$

Дефинираме случайната сума

$$\overline{S} = \sum_{n=0}^{M} k^{-n} a_n.$$

Тъй като $|k^{-n}a_n| < (\lambda/k)^n$ и $|\lambda/k| < 1$, то тази сума е равномерно ограничена за всички M. Тогава

$$E[\overline{S}] = \sum_{n=0}^{\infty} \Pr(M \geq n) k^{-n} a_n = S.$$

Тази формула води до следния Монте Карло метод за пресмятане на безкрайната сума:

$$S_N = N^{-1} \sum_{i=0}^{N} \sum_{n=0}^{M_i} k^{-n} a_n,$$

в който стойностите M_i са избрани в съответствие с горното разпределение. При този метод стойностите a_n могат също да бъдат пресметнати с Монте Карло метод.

4.3.8 Адаптивни алгоритми

Друга група алгоритми, широко използвани за числено интегриране, са адаптивните алгоритми. Повечето от адаптивните алгоритми използват редица от подобласти с намаляващ обем на дадената област, избрани така, че да се концентрират пресмятанията на подинтегралната функция в подобластите, в които има особености. Най-общо, използват се два вида стратегии за разделяне: локално и глобално разделяне. Основният недостатък на локалното разделяне е, че се изисква локална абсолютна точност, която може да се определи след удовлетворяването на глобалната абсолютна точност. Основното предимство на стратегията за локално разделяне е лесната процедура на обработка ма подобластите (не се съхранява информация за неактивните подобласти). Обикновено глобалните адаптивни алгоритми използват повече работна памет, отколкото локалните, и определянето на извадката става по-бавно. Тези алгоритми целят минимизиране на глобалната грешка, колкото е възможно по-бързо, независимо от определеното изискване за точност.

При обикновения адаптивен Монте Карло метод не се изисква допълнителна гладкост на подинтегралната функция. При този метод се генерират N равномерно разпределени случайни точки в s-мерния единичен

куб. За генерирането на всяка точка, както обикновено, се генерират s равномерно разпределени случайни числа в интервала $[0,1]$. Алгоритъмът е адаптивен: започва се с някакво сравнително малко число N, което се задава като входна данна. В процеса на пресмятането се оценява вариацията по всяка от координатните оси и тази информация се използва за увеличаване на плътността на новогенерираните случайни точки, При този метод вместо стандартната оценка за вероятната грешка $\varepsilon \leq 0.6745\sigma(\theta)\frac{1}{\sqrt{N}}$, ще имаме $\varepsilon \leq c\frac{1}{\sqrt{N}}$, където очевидно $c \leq 0.6745\sigma(\theta)$ поради адаптивността на алгоритъма.

При ускорения адаптивен метод областта на интегриране първоначално се разделя на еднакви по обем подобласти. За всяка от подобластите се пресмята приближено интеграла и се прави апостериорна оценка на дисперсията. След това се получава едно приближение на интеграла в цялата област и за тоталната дисперсия. Последната се сравнява с локалните апостериорни оценки за дисперсията във всяка от подобластите и тази информация се използва за по-нататъшно разделяне на областта и увеличаване плътността на случайните точки. При този метод вероятната грешка има вида $\varepsilon \leq cN^{-1/2-1/s}$.

Очевидно е, че при големи размерности s сходимостта асимптотично клони към $O(N^{-1/2})$, а поради по-високата изчислителна трудоемкост на този метод, при големи s той дава по-лоши резултати.

Глава 5

Квази-Монте Карло методи

5.1 Неравенство на Коксма-Хлавка

Монте Карло методите се основават на използването на псевдослучайни числа, а квази-Монте Карло методите се основават на детерминистични редици с малък дискрепанс (наричани квазислучайни редици), [14]. Докато псевдослучайните генератори са конструирани, така че да имитират поведението на истински случайните числа, квазислучайните редици са построени така, че да са толкова равномерно разпределени, колкото е математически възможно.

Разглеждаме задачата за приближено пресмятане на s-мерен интеграл:

$$I(f) = \int_{I^s} f(x)dx = \int_0^1 \dots \int_0^1 f(x^{(1)}, \dots, x^{(s)})dx^{(1)} \dots x^{(s)}$$

чрез сумата

$$Q_N(f) = \sum_{i=1}^{N} w_i f(x_i),$$

където $x_i \in [0,1)^s$ и w_i са тегла. Ако $\{x_1, \dots, x_N\}$ са избрани случайно, и $w_i = \frac{1}{N}$, то $Q_N(f)$ е стандартното Монте Карло приближение, чиято статистическа грешка се оценява с използване на Централната гранична теорема. Ако $\{x_1, \dots, x_N\}$ е множество от квазислучайни числа, то $Q_N(f)$ е квази-Монте Карло оценка. В този случай, точността се оценява от неравенството на Коксма-Хлавка:

Теорема 5.1.1 (Коксма-Хлавка) *За всяка редица* $\{x_i\}_{1\leq i\leq N}$ *и всяка функция* $f(x)$ *с ограничена в смисъл на Харди-Краус вариация* $V(f)$, *е изпълнено:*

$$\left|\frac{1}{N}\sum_{i=1}^{N} f(x_i) - I(f)\right| \leq V(f)D_N^{\star}.$$

където $D_N^{\star}$ *е дискрепанс-звезда на точковото множество* $\{x_i\}_{1\leq i\leq N}$.

За този фундаментален резултат в теорията на квази-Монте Карло методите, Кафлиш [14] предлага изненадващо просто и елегантно доказателство на неравенството (за случая на функции с ограничени s производни), което илюстрира изключителната му важност в теорията на квази-Монте Карло методите. Доказателството, изложено по-долу, се базира на статията на Кафлиш [14].

Да дефинираме вариация на функция на една променлива, $f(x)$, като

$$V(f) = \int_0^1 \left|\frac{df}{dx}\right| dx.$$

За размерност s, вариация в смисъл на Харди-Краус (за функции с ограничени s-ти производни) се дефинира като:

$$V(f) = \int_{[0,1)^s} \left|\frac{\partial^s f}{\partial x^{(1)}\ldots\partial x^{(s)}}\right| dx^{(1)}\ldots dx^{(s)} + \sum_{j=1}^{s} V(f_1^{(j)}),$$

където $f_1^{(j)}$ е рестрикцията на f върху границата $x_j = 1$.

Ако използваме означението $R(J(v))$, въведено от Кафлиш, за редици с N точки $\{x_i\}$ в единичния хиперкуб $I^s = [0,1)^s$ може да се дефинира

$$R_N(J(v)) = \prod_{j=1}^{s} v_j - \frac{1}{N}\#\{x_i \in J(v)\}, \tag{5.1.1}$$

и

$$D_N^{\star} = \sup_{v\in I^s} |R_N(J(v))|.$$

Да отбележим, че в единичния хиперкуб, $I^s = [0,1)^s$,

$$R(x) = \left[1 - \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)\right] dx,$$

където съгласно дефиницията в равенство (5.1.1), $R(x) = R_N(J(x))$, и да разгледаме функция f, която приема нулеви стойности по границата на единичния куб I^s. Тогава

$$\left|I(f) - \frac{1}{N}\sum_{i=1}^{N} f(x_i)\right| = \left|\int_{I^s} f(x)dx - \frac{1}{N}\sum_{i=1}^{N} f(x_i)\right|$$

$$= \left|\int_{I^s} \left[1 - \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)\right] f(x)dx\right|$$

$$= \left|\int_{I^s} R(x)df(x)\right|$$

$$\leq (\sup_x R(x)) \int_{I^s} |df(x)|$$

$$= D_N^* V(f).$$

За функции f, които са ненулеви върху границата на единичния куб, членовете при интегрирането по части са ограничени от граничните членове $V(f)$.

Трябва да се отбележи, че неравенството на Коксма-Хлавка дава теоретична оценка, която е трудно да се използва при реални пресмятания, поради трудностите при изчисляване на вариацията на функцията и на дискрепанс-звезда на редицата.

Да сравним грешката при интегриране с Монте Карло и квази-Монте Карло метод:

- И в двата случая грешката представлява произведение от два множителя, първият от които зависи само от подинтегралната функция, а вторият – само от редицата. При това, порядъка на грешката е $N^{-1/2}$ при интегриране с Монте Карло метод, и $(\log N)^s N^{-1}$ с квази-Монте Карло метод.
- При Монте Карло грешката има вероятностен характер, докато квази-Монте Карло дава грешка в най-лошия случай (неравенство).

Класическото обобщение на интегриране чрез квази-Монте Карло метод е да се разгледа сумата $\frac{1}{N}\sum_{n=1}^{N} f(\alpha_n)$, където редицата $\alpha = \{\alpha\}$ е

извадка от не-равномерно разпределение. Нека $g(x)$ е функцията на вероятностна плътност и $G(x)$ е функцията на разпределение. Тогава G-звезда дискрепанс на точките $\alpha_1, \dots \alpha_N$ се дефинира като:

$$D_N^\star(\alpha_n; G) = \sup_{x \in [0,1]^s} \left| \frac{1}{N} \sum_{n=1}^{N} 1_{[0,1)}(\alpha_n - G(x)) \right|.$$

Редицата α се нарича g-разпределена, ако $D_N^\star(\alpha_n; G) \to 0$ при $N \to \infty$. В сила е следното твърдение [15]:

Теорема 5.1.2 *Нека f е функция с ограничена вариация в смисъл на Харди и Краус върху $[0,1)^s$ и $\alpha = \{\alpha\} \in [0,1)^s$. Нека g е вероятностна плътност с функция на разпределение G върху $[0,1]^s$. Тогава за всяко $N > 0$*

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(\alpha_n) - \int f(x) dG(x) \right| \leq V_{HK}(f) D_N^\star(\alpha_n; G).$$

Друг начин за обобщаване на квази-Монте Карло правилото за интегриране е да се определят "тегла"на възлите $\alpha_1, \dots \alpha_N$ и да се разгледат формули за интегриране с тегла

$$\frac{1}{P(N)} \sum_{n=1}^{N} p_n f(\alpha_n)$$

за оценка на $\int_{[0,1]^s} f(x) dx$. Нека $p = (p_n)$ е редица от реални числа и $P(N) = \sum_{n=1}^{N} p_n$. Дискрепансът на редицата $\alpha_1, \dots \alpha_N$ по отношение на тегловната редица p се дефинира като

$$D_N^\star(\alpha_n; p) = \sup_{x \in [0,1]^s} \left| \frac{1}{P(N)} \sum_{n=1}^{N} p_\alpha 1_{[0,x)}(\alpha_n) - x_1 \dots x_s \right|.$$

Неравенство от типа на Коксма-Хлавка за претеглени усреднени суми, когато теглата са положителни, е дадено в Нидерайтер и Тихи [49]:

Теорема 5.1.3 *Нека f е фукция с ограничена вариация в смисъл на Харди и Краус върху $[0,1]^s$, $\alpha = (\alpha_n) \in [0,1)^s$ и $p = (p_n)$ е тегловната редица. Тогава за всяко $N > 0$*

$$\left| \frac{1}{P(N)} \sum_{n=1}^{N} p_\alpha f(\alpha_n) - \int f(x) dx \right| \leq V_{HK}(f) D_N^\star(\alpha_n; G).$$

Доказателствата на горните две теореми предствляват обобщение на доказателството на Заремба в [70].

Двете обобщения на класическата квази-Монте Карло квадратура могат да се комбинират, за да се получат „претеглени не-равномерни квази-Монте Карло квадратури“, т.е., квази-МК правило за интегриране с тегла:

$$\frac{1}{P(N)}\sum_{n=1}^{N} p_n f(\alpha_n),$$

където редицата $\alpha = (\alpha_n)$ е извадка от неравномерно разпределение. Съответният дискрепанс е:

$$D_N^\star(\alpha_n; G, p) = \sup_{x\in[0,1]^s} \left| \frac{1}{P(N)}\sum_{n=1}^{N} p_n 1_{[0,x)}(\alpha_n) - G(x) \right|.$$

5.2 ANOVA декомпозиция и ефективна размерност

Да отбележим, че порядъкът на грешката на квази-Монте Карло метод е *асимптотически* по-добър от този на съответния Монте Карло метод. Така, за фиксирано s и достатъчно голямо N, квази-Монте Карло квадратурата превъзхожда Монте Карло квадратурата. Обаче, когато s е голямо, факторът $N^{-1}(\log N)^s$ е съществено по-голям от $N^{-1/2}$, освен ако N е огромно число. Освен това, да забележим, че функцията $kN^{-1}(\log N)^s$ при k произволна константа има максимум за $N = e^s$. Със сигурност, асимптотичният режим не е достигнат преди тази стойност на N. По тази причина дълго време беше широко разпространено схващането, че квази-Монте Карло методът не бива да се използва за интегриране при висока размерност, например $s \geq 15$. Концепцията *ефективна размерност*, която ще разгледаме по-долу, помага за обяснението защо квази-МК продължават да са ефективни и за задачи с голяма размерност.

И така, ефективната размерност е важна концепция при квази-МК интегрирането. За да я дефинираме, първо ще разгледаме ANOVA (analysis-of-variance) декомпозиция на функция.

Декомпозицията ANOVA е начин за декомпозиция на функция като сума от по-прости функции. Нека $S = \{1, \ldots, s\}$. За всяко подмножество $u \subseteq S$, нека $|u|$ е кардиналното му число, а $S\backslash u$ е неговото допълващо (complimentary) множество в S. Нека x_u е $|u|$-мерният вектор, съдър-

жащ координатите на x с индекси в u. По-нататък, нека $[0,1)^u$ означава u-мерният единичен куб, включващ координатите на u.

Да предположим, че $f(x)$ е функция с интегруем квадрат. Можем да запишем $f(x)$ като сума от 2^s функции (наричани „ANOVA членове“), един за всяко подмножество на S, а именно

$$f(x) = \sum_{u \subseteq S} f_u(x). \tag{5.2.1}$$

ANOVA членовете $f_u(x)$ се дефинират рекурсивно чрез:

$$\begin{aligned} f_u(x) &= \int_{[0,1)^{S\setminus u}} \left(f(x) - \sum_{v \subset u} f_v(x) \right) d_{S\setminus u} \\ &= \int_{[0,1)^{S\setminus u}} (f(x) d_{S\setminus u}) - \sum_{v \subseteq S} f_v(x). \end{aligned} \tag{5.2.2}$$

ANOVA членът $f_u(x)$ е тази част от функцията, която зависи само от променливите x_j при $j \in u$. От тази дефиниция следва, че константният ANOVA член за $u = \emptyset$ е точно интегралът от тази функция,

$$f_\emptyset = \int_{[0,1)^s} f(x) dx.$$

ANOVA членовете имат следните интересни свойства:

- от декомпозицията в (5.2.2) може да се покаже по индукция по $|u|$, че
$$\int_0^1 f_u(x) dx_j = 0 \text{ за } j \in u.$$
- Различните членове на ANOVA декомпозицията са ортогонални в смисъл, че
$$\int_{[0,1)^s} f_u(x) f_v(x) dx = 0, \quad \text{за} \quad u \neq v.$$
- Ако $\sigma^2(f)$ е дисперсията на f, то от ортогоналността на ANOVA членовете и използването на декомпозицията (5.2.1), тази дисперсия може да се представи като
$$\sigma^2(f) = \sum_{u \subseteq S} \sigma_u^2(f),$$

където $\sigma_u^2(f) = \int_{[0,1)^s}(f_u(x))^2 dx$ за $|u| > 0$ е дисперсията на f_u и $\sigma_\emptyset^2(f) = 0$.

Чрез сравняване на дисперсията на функцията f със сумата от вариациите върху специално избрани подмножества на S от ANOVA декомпозицията f, ефективната размерност може да се дефинира по два начина.

Дефиниция 5.2.1 *Ефективната размерност на f в смисъл на суперпозиция (или суперпозиционна размерност) е най-малкото цяло число d_s, такова че*

$$\sum_{0<|u|\leq d_s} \sigma_u^2(f) \geq p\sigma^2(f),$$

където $0 < p < 1$ (обикновено p е близо до 1).

Суперпозиционната размерност е индикатор дали членовете от нисък ред доминират функцията. Това е особено полезно в случая, когато всички променливи са еднакво важни или почти еднакво важни.

Дефиниция 5.2.2 *Ефективната размерност на f в смисъл на отсеченост (truncation sense) (или „отсечена размерност“) е най-малкото число d_t, такова че*

$$\sum_{u\subseteq\{1,\ldots,d_t\}} \sigma_u^2(f) \geq p\sigma^2(f).$$

Грубо казано, отсечената размерност е броя на „важните“ променливи и индикира на колко и кои променливи ние трябва да обърнем внимание. Тя е особено подходяща за характеризация на „тегловните“ функции, където някои променливи са по-важни от други.

Ефективната размерност зависи от p. За едно и също p винаги имаме $d_s \leq d_t \leq s$. Да отбележим, че обратното не е вярно. Например, функцията $f(x) = x_1 + \cdots + x_s$ има суперпозиционна размерност 1, но може да има много по-голяма отсечена размерност. Освен това, d_s не зависи от реда, по който са индексирани входните променливи, но d_t зависи.

5.3 Гладкост и намаляване на размерността

Загубата на теоретично очакваната точност при интегриране по квази-Монте Карло метод се дължи на две главни причини:

- Прекъснатост или липса на гладкост на подинтегралната функция. Оценката на грешката $O(N^{-1}\log^s N)$ се определя от неравенството на Коксма-Хлавка, което е в сила за функции с ограничена вариация $V(f)$. Но за да бъде $V(f)$ крайна, f трябва да бъде гладка. От друга страна, на практика Монте Карло методите често се използват за пресмятане на интеграли от прекъснати функции. Например, много методи включват процес на вземане на решения, при което компонент на подинтегралната функция е 1 или 0, съответно ако решението е „да“ или „не“. За такава прекъсната подинтегрална функция не може да се приложи неравенството на Коксма-Хлавка и редица числени експерименти показват грешка с порядък $O(N^{-1/2})$.

- Висока размерност. За големи s дискрепансът на квазислучайната редица се доближава до $O(N^{-1/2})$ при средно големи стойности на N, но остава $O(N^{-1}\log^s N)$ за достатъчно големи N. Преходът от $O(N^{-1/2})$ към $O(N^{-1}\log^s N)$ се осъществява приблизително при $N = e^s$.

Целта на този раздел е да се покаже как тези две основни ограничения за приложимостта на квази-Монте Карло методите могат да бъдат преодолени. Това се постига чрез модификации на съответните Монте Карло методи – изглаждане на прекъснатостта и намаляване на ефективната размерност.

Стандартният метод на селекцията, използван за получаване на извадка с плътност, пропорционална на подинтегралната функция, в метода на съществената извадка, може да се перефразира като интегриране на прекъсната функция (прекъсването се дължи на решението да се приеме или отхвърли генерираната точка). Това води до по-голяма грешка в съответния квази-Монте Карло метод. За да се преодолее този проблем, ще формулираме *метод на селекция с изглаждане*, при който се запазва порядъка на сходимост $O(N^{-1}\log^s N)$.

Да припомним, че в метода на съществената извадка за оценка на интеграла $I = \int_D f(x)dx$ въвеждаме функция h, която имитира поведението на f върху D, и която е лесно интегруема аналитично или числено. Процедурата за генериране на извадка се променя, като се генерират точки, разпределени с плътност h, вместо равномерно разпределени точки. Съответно, вместо да оценяваме $f(x)$ във всяка точка от извадката, ние оценяяваме $f(x)/h(x)$, което води до неизместена оценка на интеграла, тъй

като

$$E_h\left(\frac{f}{h}\right) = \int_D \frac{f(x)}{h(x)}.h(x)dx = \int_D f(x)dx = I.$$

Съответната Монте Карло оценка може да бъде записана като:

$$\widetilde{I_N} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{h(x_i)}, \quad x_i \sim h(x) \tag{5.3.1}$$

където $x_i \sim h(x)$ означава, че точките от извадката $x_1, \ldots, x_N$ имат вероятностна плътност h (h трябва да е нормализирана). Дисперсията на тази оценка е съществено по-малка, ако $f(x)/h(x)$ има по-малка дисперсия от f.

За ограничен клас от плътности точките от извадката могат да се генерират чрез метода на обратната функция. В общия случай, обаче, трябва да се използва по-обща процедура, каквато е методът на селекцията.

Стандартен метод на селекцията. По-долу следва аалгоритъма на селекцията, записан във вид, удобен за влагане в метода на съществената извадка. Добавена е допълнителна размерност (променливата y, която се използва за вземане на решение).

1. Избираме $\gamma \geq \sup_{x\in D} h(x)$.
2. Повтаряме, докато получим N точки:
   (а) Генерираме (x_t, y_t) от стандартно равномерно разпределение.
   (б) Ако $y_t < \gamma^{-1}h(x_t)$ приемаме точка x_t иначе я отхвърляме.

Алгоритъмът произвежда редица от точки в размерност s (редицата съдържа приетите точки), които са разпределени съответно на плътността h, както се изисква в метода на съществената извадка.

Сумата (5.3.1), в която точките x_t са избрани по метода на селекцията, може да се интерпретира като Монте Карло оценка на следния интеграл:

$$I = \gamma\int_D\int_0^1 \frac{f(x)}{h(x)}\chi(y < \gamma^{-1}h(x))dydx \tag{5.3.2}$$

Директната Монте Карло оценка на този интеграл е

$$\widetilde{I_N} = \frac{\gamma}{N}\sum_{i=1}^{N}\chi\left\{y_t < \frac{h(x_t)}{\gamma}\right\}.\frac{\widetilde{f}(x_t)}{h(x_t)} \tag{5.3.3}$$

Дисперсията на този интеграл е редуцирана чрез заместване на общия брой на точките, N, със приетите точки, за да се получи следната Монте Карло оценка:

$$\widetilde{I_N} = \frac{1}{\sum_{i=1}^{N} \chi\left\{y_t < \frac{h(x_t)}{\gamma}\right\}} \sum_{i=1}^{N} \chi\left\{y_t < \frac{h(x_t)}{\gamma}\right\} \frac{f(x_t)}{h(x_t)} \tag{5.3.4}$$

Една разлика между сумата (5.3.4) и пряката Монте Карло оценка в (5.3.2) е, че в (5.3.4) броят N е броят на приетите точки, а не броят на всички опити. Това се отразява в множителя γ в (5.3.3).

Квази-Монте Карло методът не може да се приложи ефективно за пресмятане на интеграла (5.3.3), защото подинтегралната функция съдържа характеристична функция, отнасяща се до решението за приемане или отхвърляне.

Метод на селекцията с изглаждане. Методът на селекцията води до прекъсване на подинтегралната функция в резултат на самата природа на двоичното решение: приемане или отхвърляне. Това може да се преодолее, като се разреши точките от извадката да имат асоциирани тегла за приемане: Ще формулираме метод на селекцията с изглаждане, който има предимствата на метода на селекцията, но не предизвиква прекъсване. Първо, заместваме интеграла от (5.3.2) с еквивалентен интеграл:

$$I = \gamma \int_D \int_0^1 \frac{f(x)}{h(x)} \chi_\delta(y, \gamma^{-1} h(x)) dy dx \tag{5.3.5}$$

в който гладката функция χ_δ удовлетворява

$$\int_0^1 \chi_\delta(y, \gamma^{-1} h(x)) dy = \gamma^{-1} h(x)). \tag{5.3.6}$$

Функцията χ_δ, която ще наричаме още тегловна функция w, ще изберем да бъде частично линейна.

Съответната Монте Карло апроксимация съответства на следната процедура на селекция с изглаждане:

1. Избираме $\gamma \geq \sup_{x \in D} h(x)$ и $0 < \delta \ll 1$.
2. Повтаряме, докато теглото на приетите точки е N: (а) Генерираме (x_t, y_t) от стандартно равномерно разпределение в $[0,1)^{s+1}$. (б) Ако $y_t < \frac{h(x_t)}{\gamma} - \frac{1}{2}\delta$, то теглото на приемане е $w = 1$, ако $y_t > \frac{h(x_t)}{\gamma}$, то $w = 0$. В противен случай. $w = \frac{1}{\delta}(\frac{h(x_t)}{\gamma} + \frac{1}{2}\delta - y_t)$.

Плътността на приетите точки x е $f_{accept}(x)$, дадена чрез (при $w = \chi_\delta$):

$$f_{accept}(x) = \frac{1 \cdot \int_0^1 w(x,y)dy}{\int_{I^s}\left[1 \cdot \int_0^1 w(\xi,y)dy\right]d\xi} = \frac{h(x)/\gamma}{1/\gamma}$$

$$f_{accept}(x) = h(x)$$

което показва, че сме генерирали точки с плътност h.

При този метод има допълнителна работа в сравнение с оригиналния метод на селекцията. Първо, това са операциите, свързани с присвояването на тегло на всяка точка и съхраняването и използването на тези тегла. Обикновено обаче, тези разходи са твърде малки и могат да се разглеждат като несъществени. Второ, има допълнителна работа, свързана с приемането на точките с тегла, по-малки от 1: т.е., за да постигнем общо тегло на приемане с размер N, необходими са повече от N оценявания. Тази допълнителна работа може да се минимизира чрез определяне на константата δ да бъде сравнително малка. От друга страна, ако δ е прекалено малка, ще се загубят предимствата на непрекъснатостта.

Претеглена равномерна извадка. Друга алтернатива на метода на селекцията може да се формулира като изцяло се елиминира решението приемане/отхвърляне, и вместо това, на всяка точка се присвоява тегло, равно на нейната вероятност за приемане, дадена чрез $\gamma^{-1}h(x_t)$.

Новата Монте Карло оценка се получава, както следва:

$$\widetilde{I_N} = \frac{\sum_{i=1}^N \frac{f(x_t)}{h(x_t)} \cdot \frac{h(x_t)}{\gamma}}{\sum_{i=1}^N \frac{h(x_t)}{\gamma}} = \frac{\sum_{i=1}^N f(x_t)}{\sum_{i=1}^N h(x_t)}$$

Тази сума всъщност е отношение на две оценки от обикновения Монте Карло метод за интегриране - оценка на оригиналната функция f в числителя, и оценка на функцията за съществената извадка h в знаменателя. Получената в резултат оценка е изместена, но е доказано (Powell и Swann през 1996, както и Spanier, Maize през 1994), че изместването е незначително в сравнение с грешката при $N \to \infty$. Нещо повече, положителната корелация между f и h, когато h е добре избрана да имитира поведението на f, води до съществено намаляване на дисперсията.

Отклонението и средноквадратичната грешка са следните:

$$\text{bias}(\widetilde{I_N}) = \frac{I\text{var}(h)}{N} - \frac{\text{cov}(f,h)}{N} + O(N^{-3/2}) \qquad (5.3.7)$$

$$\mathrm{rmse}(\widetilde{I_N}) = \frac{\sqrt{\mathrm{var}(f) + I^2\mathrm{var}(h) - 2I\mathrm{cov}(f,h)}}{\sqrt{N}} + O(N^{-3/4}) \qquad (5.3.8)$$

Едно от предимствата на метода на претеглената равномерна извадка, споменато още от Powel и Swann (1966), е че не трябва да се генерира извадка за плътността h. Другото предимство е, че оценката по метода на претеглената извадка е непрекъсната (при предположение, че f и h са непрекъснати), така че квазислучайните числа могат ефективно да се използват в сумирането.

От друга страна, недостатък на метода на претеглената извадка е че за задачи с големи области с малко значение, много повече пресмятания на функцията ще бъдат извършени в такива области, отколкото ако се използва метода на съществената извадка. Това се отразява на по-голямото намаление на дисперсията при съществената извадка; за такива задачи е по-подходящ метода на съществената извадка с изглаждане.

Глава 6

Паралелни пресмятания

Алтернативен начин да се подобри оценката на решението, получена с Монте Карло или квази-Монте Карло методите, са паралелните пресмятания. Ако n процесора изпълняват n независими еднакви копия на дадено приложение, акумулираният резултат ще притежава n пъти по-малка дисперсия. Поради естеството на тези методи, за някои задачи е подходящо да се пресметнат, колкото е възможно повече копия. От друга страна, след стартирането на една разпределена задача, тя обикновено се изпълнява почти без никакви комуникации между отделните подзадачи. По тази причина широко разпространено е схващането, че Монте Карло и квази-Монте Карло методите са изчислително сложни, но естествено паралелни. Появата на все по-мощни компютри, особено на паралелни и разпределени системи, направи възможно извършването на все по-амбициозни пресмятания с разпределени Монте Карло приложения. Една друга възможност за пресмятания в разпределена среда са грид технологиите. Изчислителният грид е сравнително нов тип изчислителна инфраструктура, която осигурява достъп до изчислителни ресурси, разпределени по цял свят. В резултат на високия си капацитет, тази мощна международна инфраструктура значително превъзхожда възможностите на локални клъстери и отделни изчислителни центрове, като предоставя уникална възможност за научни изследвания, изискващи големи изчислителни ресурси. Монте Карло приложенията са едни от първите и най-разпространени грид приложения.

Трябва да се отбележи, че необходимо условие за успешната работа на

паралелните и грид Монте Карло приложения е подходящото разпаралеляване на използваните случайни числа. Библиотеката SPRNG (Scalable Parallel Random Number Generators) е специално създадена за генериране на независими случайни потоци чрез параметризация на псевдослучайни генератори.

Техниките, използвани от паралелните генератори на случайни числа за разпределяне на последователно генерирани редици между различните процесори, включват

- разделяне на блокове: редицата се разделя чрез разцепване на непресичащи се непрекъснати отрязъци. Ако дължината на всеки отрязък е L в случайната редица $\{X_n\}$, то процесорът P_i получава случайна редица с числа:
$$X_{iL}, X_{iL+1}, X_{iL+2}, \ldots,$$
- „жабешки скок“: раздаване на числата на отделните процесори, както се раздава тесте карти между играчите при игра на карти: ако скокът е L и редицата е $\{X_n\}$, то процесор P_i получава числата
$$X_i, X_{i+L}, X_{i+2L}, \ldots,$$
- параметризация: началното ядро или други параметри на избрания генератор се избират внимателно така, че да се генерират независими редици с голям период за всеки процесор. Например, за линейния конгруентен генератор, който се представя с формулата
$$X_{n+1} = aX_n + b \pmod{p},$$
има два начина за параметризация на този генератор. Единият е избор на различни адитивни константи b за формиране на различни редици по модул степен на две, другият е избор на различни множители, a, за редици по модул просто число.

Основният начин за генериране на паралелни редици от случайни числа е конструирането на независими редици чрез подходящо параметризиране на псевдослучайните генератори. Другите два начина за разпаралеляване понякога имат корелации. Този факт трябва да се има пред вид, когато се избира размера на блоковете или скока. Освен това, периодът на началния генератор трябва да бъде достатъчно дълъг, така че да има достатъчно много случайни числа, които да бъдат разпределени към отделните процесори.

По принцип, трудно е да се намерят добри генератори на псевдослучайни числа, а още по-трудно е да се намерят ефективни алгоритми за генерация на псевдослучайни числа върху паралелни компютри. Само няколко са достъпните софтуерни пакети за генератори на псевдослучайни числа, такива като библиотеката SPRNG, Mersenne Twister, и нелинейния обратен конгруентен генератор от pLab.

Още по-сложен е въпросът с паралелизацията на квазислучайните редици. Както при методите за разпаралеляване на псевдослучайни редици, съществуват същите три основни начина за разпаралеляване на квазислучайни редици. При първия и втория подход се произвеждат числа от една квазислучайна редица. Третият подход изисква фамилия от квазислучайни редици. Такава фамилия може да се генерира като се приложат разбъркващи техники – с използване на пермутации или с използване на псевдослучайни числа. Различни пермутации ще доведат до различни редици. Всеки разбъркан вариант на изходната редици може да се разглежда като друг паралелен поток от квазислучайни числа.

Има два основни начина за конструиране на разбъркани квазислучайни редици. Първият метод се основава на рандомизирано изместване във формата:

$$z_n = x_n + r \pmod 1,$$

където x_n е квазислучайно число в $[0,1)^s$, а r е s-мерно псевдослучайно число. Използвайки различни псевдослучайни числа r получаваме различни разбъркани версии $\{z_n\}$ на оригиналната квазислучайна редица $\{x_n\}$. Вторият метод се основава на цифрови пермутации. Съществуват различни методи за разбъркване, основани на цифрови пермутации, като разликите в методите се основават на различни дефиниции на пермутациите, например методите на Оуен, Тезука и Матошек. Важно е да се подчертае, че при всички методи се използват псевдослучайни числа за разбъркването, т.е., важно е да се използва добър генератор на псевдослучайни числа. Освен това, тъй като основното приложение на разбърканите редици са паралелните пресмятания, използваният генератор е желателно да бъде от библиотека паралелни генератори.

Паралелните квази-Монте Карло методи изискват квазислучайни редици, разпределени между отделните процесори. За разлика от изследванията на приложенията на паралелните псевдослучайни генератори, публикациите относно използването на квазислучайни редици в алгоритми за паралелни пресмятания са сравнително малко. Идеята при първите два

подхода е: съответните подредици, с които работят отделните процесори, да имат същата равномерност, както изходната редица. Третият подход е по-удобен за работа и в някои случаи може да се докаже, че съответните разбъркани редици имат същата равномерност, както редицата, от която са получени.

Литература

[1] Б. Боянов, *Лекции по числени методи*, Дарба, София, 1998.

[2] С. Ермаков и Г. Михайлов, *Статистическое моделирование*, Наука, Москва, 1982.

[3] С. Никольский, *Квадратурни формули*, Наука, Москва, 1988 (на руски език).

[4] И. Собол, *Числени методи*, Наука, Москва, 1973 (на руски език).

[5] Н. Янев и Б. Димитров, *Вероятности и статистика*, Наука и изкуство, София, 1990.

[6] I. A. Antonov and V. M. Saleev, An economic method of computing LPτ-sequences, *USSR Computational Mathematics and Mathematical Physics*, **19**:252–256, 1979.

[7] E. Atanassov, A New Efficient Algorithm for Generating the Scrambled Sobol Sequence, Numerical Methods and Applications, Springer, *LNCS* **2542**:83–90, 2003.

[8] E. Atanassov, On the discrepancy of the Halton sequences, *Mathematica Balkanica. New Series*, **18(1–2)**:15–32, 2004.

[9] E. Atanassov and M. Durchova, Generating and testing the modified Halton sequences, Springer-Verlag Heidelberg, *Lecture Notes in Computer Science*, **2542**:91–98, Berlin, 2003.

[10] Eric Braaten and George Weller, An improved low-discrepancy sequence for multidimensional quasi-Monte Carlo integration, *Journal of Computational Physics*, **33**:249–258, 1979.

[11] P. Bratley and B. L. Fox, Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator, *ACM Transactions on Mathematical Software*, **14(1)**:88–100, 1988.

[12] B. C. Bromley, Quasirandom Number Generation for Parallel Monte Carlo Algorithms, *Journal of Parallel Distributed Computing*, **38(1)**:101–104, 1996.

[13] J. M. Bull and T. L. Freeman, Parallel algorithms for multi-dimensional integration, *Parallel and Distributed Computing Practices*, **1(1)**:89–102, 1998.

[14] R. E. Caflisch, Monte Carlo and quasi-Monte Carlo methods, *Acta Numerica*, **7**:1–49, 1998.

[15] P. Chelson, „*Quasi-Random Techniques for Monte Carlo Methods*“, PhD dissertation, Claremont Graduate School, 1976.

[16] H. Chi and M. Mascagni, „Efficient Generation of Parallel Quasirandom Sequences via Scrambling,“ Springer *LNCS*, **4487**:723–730, 2007.

[17] R. Cranley and T. N. L. Patterson. Randomization of number theoretic methods for multiple integration. S*IAM Journal on Numerical Analysis*, **13(6)**: 904–914, 1976.

[18] Ivan Dimov, Monte Carlo for Applied Scientists, World Scientific, 2008.

[19] I. Dimov, A. Karaivanova, Overconvergent Monte Carlo Methods for Density-function Modelling Using B-Splines, *Advances in Numerical Methods and Appl.*, World Scientific, 1994, 85–93.

[20] Ivan Dimov, *Monte Carlo for Applied Scientists*, World Scientific, 2008.

[21] P. L' Ecuyer, Efficient and portable combined random number generators, *Communications of the ACM*, **31**:742–749, 1988.

[22] H. Faure, Discrépance de suites associées à un système de numération (en dimension s), *Acta Arithmetica*, **XLI**, 1982, 337–351.

[23] H. Faure, Monte Carlo and Quasi-Monte Carlo Methods for Numerical Integration, *Combinatorial and computational mathematics*, **1**:1–12, 2001.

[24] H. Faure, Variations on (0; s)-sequences, *Journal of Complexity*, **17(4)**:741–753, 2001.

[25] B. Fox, Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator, *ACM Transactions on Methematical Software*, **14(1)**:88–100, 1988.

[26] John H. Halton, On the efficiency of certain quasi-random sequences of points in evaluating multidimensional integrals, *Numerische Mathematik*, **2**:84–90, 1960.

[27] J. M. Hammersley, D. C. Handscomb, *Monte Carlo Methods*, Jonh Wiley and Sons, inc., New York, London, Methuen, 1964.

[28] T. Hesterberg, Weighted average importance sampling and defensive mixture distributions, *Technometrics*, **37(2)**:185–194, 1995.

[29] S. Ivanovska, A. Karaivanova, Parallel Importance Separation for Multiple Integrals and Integral Equations, *Computational Science (Bubak, Dick van Albada, Sloot, Dongara Eds.)*, Springer LNCS **3039**:499–506, 2004.

[30] S. Ivanovska, E. Atanassov, A. Karaivanova, A Superconvergent Monte Carlo Method for Multiple Integrals on the Grid, *LNCS*, **3516**, Springer-Verlag, Berlin-Heidelberg, 2005, 735–742.

[31] H. Kahn, Random sampling (Monte Carlo) techniques in neutron attenuation problems, *Nucleonics*, **6(5)**:27–30, 1950; **6(6)**:60–65, 1950.

[32] Malvin H. Kalos and Paula A. Whitlock, *Monte Carlo Methods, volume I: basics*, John Wiley and Sons, 1986.

[33] M. Kalos and P. Whitlock, *Monte Carlo Methods*, Second revised and enlarged edition, WILEY-VCH GmbH&Co. KGaA, Meinheim, 2008, ISSN: 978-3-527-40760-6.

[34] A. Karaivanova, Adaptive Monte Carlo methods for numerical integration, *Mathematica Balkanica*, **11**:391–406, 1997.

[35] A. Karaivanova, I. Dimov, Error Analysis of an Adaptive Monte Carlo Method for Numerical Integration, *Mathematics and Computers in Simulation*, Elsevier, **47**:201–203, 1998.

[36] A. Karaivanova, I. Dimov, S. Ivanovska, A Quasi-Monte Carlo Metod for Integration with Improved Convergence, *LNCS*, **2179**, Springer-Verlag, Berlin-Heidelberg, 2001, 158–165.

[37] Y. Li and M. Mascagni, Grid-based Monte Carlo Application, *Proceedings of Grid Computing-GRID 2002*, LNCS **2536**:13–24, 2002.

[38] N. L. Manev, Strictly Balanced and Generating Permutation Sequences, *Comptes rendus de l'Academie bulgare des Sciences*, **58, No. 2**:123–128, 2005.

[39] N. L. Manev, Sequences generating permutations, *Applied Mathematics and Computations*, Elsevier, **216, No.3**:708–718, 2010.

[40] W. L. Martinez, A. R. Martinez, *Computational Statistics Handbook with MATLAB*, Chapman and Hall/CRC, 2002.

[41] M. Mascagni and H. Chi, On the Scrambled Halton Sequence, *Monte Carlo Methods and Applications*, **10(3–4)**:435–442, 2004.

[42] J. Matousek. On the L2-discrepancy for anchored boxes. *Journal of Complexity*, **14**:527–556, 1998.

[43] M. Matsumoto and T. Nishimura, Mersenne Twister: a 623-Dimensionnally Equidistributed Uniform Pseudorandom Number Generator, *ACM Trans. Model. Comput.Simul.*, **8(1)**:3–30, 1998.

[44] N. Metropolis and S. Ulam, The Monte Carlo Method, *J. of American Statistical Association*, **44(247)**:335–341, 1949.

[45] G. A. Mikhailov, *Optimization of the "weight" Monte Carlo methods*, Moskow, 1987 (in Russian).

[46] W. J. Morokoff and R. E. Caflish, Quasi-random sequences and their discrepancies, *SIAM Journal on Scientific Computing*, **15(6)**:1251–1279, 1994.

[47] W. J. Morokoff and R. E. Caflish, Quasi-Monte Carlo integration, *Journal of Computational Physics*, **122(2)**:218–230, 1995.

[48] H. Niederreiter, *Number Generation and Quasi-Monte Carlo Methods*, SIAM Philadelphia, 1992.

[49] H. Niederreiter, R.F. Tichy. „Beitrage zur diskrepanz bezuglich gewichteter mittel", *Manuscripta Math.*, **42**:85–99, 1983.

[50] E. Novak and K. Ritter, High Dimensional Integration of Smooth Functions Over Cubes, *Numerische Mathematik*, **1**:1–19, 1996.

[51] G. Okten and A. Srinivasan, Parallel Quasi-Monte Carlo Methods on a Heterogeneous Cluster, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, Springer, 2002, 406–421.

[52] A.Owen, Scrambling Sobol and Niederreiter-Xing Points. *Journal of Complexity*, **14(4)**:466–489, 1998.

[53] A. Owen, Variance and discrepancy with alternative scramblings, *ACM Trans. On Computational Logic*, **V**:1–16, 2002.

[54] A.Owen, Variance with alternative scramblings of digital nets, *ACM Transactions on Modeling and Computer Simulation*, **13(4)**:363–378, 2003.

[55] A. Owen and Y. Zhou, Safe and effective importance sampling, Technical report, Stanford University, Statistics Department, 1999.

[56] R. Rubinstein, *Simulation and the Monte Carlo Method*, John Wiley & Sons Inc., New York, USA, 1981.

[57] W. Schmid, A. Uhl, Techniques for Parallel Quasi-Monte Carlo Integration with Digital Sequences and Associated Problems, *Mathematics and Computers in Simulation*, **55(1–3)**:249–257, 2001.

[58] I. Sloan and H. Wozniakowski, When Are Quasi-Monte Carlo Algorithms Efficient for High Dimensional Integrals?, *Jjournal of Complexity*, **14**:1–33, 1998.

[59] I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals, *USSR Computational Mathematics and Mathematical Physics*, **7**:86–112, 1967.

[60] M. Stipoevic, M. Rogina, Quantum random number generator based on photonic emission in semiconductors, *Review of Scientific Instruments*, **78**, 045104, 2007.

[61] A. Takemura, Tensor analysis of ANOVA decomposition, *J. Amer. Stat. Assoc.*, **78**:894–900, 1983.

[62] S. Tezuka, *Uniform Random Numbers: Theory and Practice*, Kluwer Academic Publishers. IBM Japan, 1995.

[63] S. Tezuka, On randomization of generalized Faure sequences, Research Report, RT0494:15 pages, 2002.

[64] S. Tezuka and H. Faure. I-binomial scrambling of digital nets and sequences, *Journal of Complexity*, **19(6)**:744–757, 2003.

[65] Bart Vandewoestyne, Quasi-Monte Carlo techniques for the approximation of high-dimensional integrals, PhD dissertation, Katholieke Universitet Leuven, Belgium.

[66] E. Veach and L. J. Guibas, Optimally combining sampling techniques for Monte Carlo rendering, *Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH '95*, 1995, 419–428.

[67] E. Veach, Robust Monte Carlo Methods for Light Transport Simulation, Ph.D. dissertation, Stanford University, 1997.

[68] Xiaoqun Wang and Fred J. Hickernell. Randomized Halton sequences, *Mathematical and Computer Modelling*, **32**:887–899, 2000.

[69] Tony T. Warnock. Computational investigations of low-discrepancy point sets. In S.K. Zaremba, editor, Applications of Number Theory to Numerical Analysis, (Proc. Sympos., Univ. Montreal, Que., 1971), 319–343. Academic Press, New York, 1972.

[70] S. K. Zaremba, The mathematical basis of Monte Carlo and quasi-Monte Carlo methods, SIAM review, **10**:303–314, 1968.

[71] S. K. Zaremba, Some applications of multidimensional integration by parts, *Annales Polonoci Mathematici*, **21**:85–96, 1968.