**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 5**

ВЕРОЯТНОСТЬ И СЛУЧАЙНЫЕ ЧИСЛА

ЦЕЛЬ РАБОТЫ

Знакомство с элементарными вероятностными распределениями и их имититация, путем генерации соответствующих случайных чисел с помощью *MATLAB*.

ВВЕДЕНИЕ

Изучение вероятностей и случайных или стохастических процессов всегда остается важной темой просто потому, что задачи реального мира содержат случайные отклонения. Такие отклонения приводят к возникновению неопределенных факторов, которые обычная математика не может описать даже приближенно. *Вероятность* события в классической теории вероятностей определяется как отношение частоты события к числу всех возможных исходов. Часто более удобно моделировать вероятность *плотностью ее распределения* как для дискретных, так и для непрерывных случайных систем. Во многих практических задачах вид функций плотностей вероятности, представляющих вероятностные распределения, известен.

## 1. Генерирование случайных чисел

Функция *MATLAB rand* генерирует псевдослучайные числа из интервала (0,1). Эти числа называются *псевдослучайными* потому, что они *выглядят* случайными последовательностями, но существует метод их воспроизводства. Последовательности генерируются детерминированным алгоритмом, но ему можно дать «затравку», которая позволит породить конкретную последовательность. Чтобы понять, как работает команда, выполните следующее:

» rand % Генерирует случайное число из (0,1)

» rand('seed',13) % Устанавливает в качестве «затравки» 13-е число

» b1 = rand(25,l) % Случайный вектор-столбец из (0,1)

» rand('seed' ,0) % Устанавливает «затравку» по умолчанию.

» А = rand(3,4) % Случайная матрица размера 3×4 из (0,1)

» rand('seed' ,13) % Опять устанавливает в качестве «затравки» 13-е число

» b2 = rand(25,l) % Воспроизводит b1

Можно «усилить» случайность, выбирая начало последовательности «случайным образом», например, связав seed с, встроенными часами компьютера. Текущее время равно

» time=round(clock) % вместо round могут быть fix, ceil или floor

и этот вектор представляет time=[Гoд Месяц День Час Минута Секунда], так что можно использовать

» rand('seed',time(6))% или rand('seed',time(5)\*time(6))

чтобы подсоединиться к ближайшей секунде, хотя, кажется, это не всегда работает: иногда лучше самому задать значение seed, предпочтительно *нечетным* числом. Хороший результат дают числа в диапазоне 10000 - 20000 (или значение 0, присваиваемое по умолчанию).

Простая *MATLAB*-функция *rand* порождает столбец из *п* равномерно распределенных на интервале (0, 1) чисел командой *x* = *rand*(*n*,1) или строку — командой *x* = *rand*(1,*n*). Аналогично, rand(*n*) порождает случайную матрицу размера *п×п,* тогда как *rand*(*m,n*) дает *т×n*-матрицу с элементами из (0,1).

Чтобы породить случайные числа из произвольного интервала *(а,b),* воспользуйтесь линейным преобразованием (*х → у*), например,

» х = rand(l,30); % 30 случайных чисел из интервала (0,1)

» а = 12; b = 99; % задаем границы интервала

» у = а + (b-а)\*х; % 30 случайных чисел из интервала (а,b)

Обратите внимание, что элементы вектора *у* теперь принадлежат интервалу (*а,b*)*.*

## 2. Случайные целые числа

Функции *MATLAB ceil, fix, floor* и *round*, вместе с *rand* порождают случайные *целые числа.* Здесь *ceil*(*x*) есть наименьшее целое ≥ *х, floor*(*х*) – это наибольшее целое ≤ *х* и *round*(*х*) ближайшее к *х* целое. Команда *fix* совпадает либо с *ceil*, либо с *floor* в зависимости от знака числа, поскольку она округляет числа до целого, ближайшего *к нулю.* В качестве примера проверьте следующее:

|  |  |  |  |
| --- | --- | --- | --- |
| *ceil*(1.2)=2, | *ceil*(3)=3, | *ceil*(1.9)=2, | *ceil*(1.5)=2, |
| *fix*(1.9)=l, | *fix*(2)=2, | *fix*(–1.6)=–l, | *fix*(-2)=–2, |
| *floor*(1.2)=l, | *floor*(3)=3, | *floor*(1.9)=l, | *floor*(1.5)=l, |
| *round*(1.2)=l, | *round*(3)=3, | *round*(1.9)=2, | *round*(-l.5)=–2 |

Можно получить столбец из *п* чисел, принимающих с равной вероятностью целые значения 0,1,..., *k* (т.е. целые значения из замкнутого интервала [0, *k*]), выполнив

» х = rand(n,l); floor((k+l)\*x)

Но мы можем получить и столбец из *п* чисел, принимающих с равной вероятностью целые значения 1,..., *k* (т.е. целые значения из замкнутого интервала [1,*k*])*,* выполнив

» ceil(k\*x)

Команда round используется немного по-другому. Например,

» round(10\*x)

порождает *п* чисел, которые принимают значения 0,1,..., 10 с неравными вероятностями. (Вы понимаете, почему? Сравните значения *х,* округленные до 0, со значениями, округленными до 1, и рассмотрите также результат *round* (9.99 *х* + 0.5).) Если нужно произвести случайную перестановку из *k* целых чисел 1,2,..., *k,* следует воспользоваться командой randperm. Например, переставьте строки матрицы случайным образом:

» r = randperm(9)

» А = ceil(rand(9)\*5) % генерирует матрицу с элементами из (1,5)

» В = А(r,:) % переставляет строки по "r"

Вот простой способ получить вектор-столбец из 0 и 1, где 0 появляется с «вероятностью 1/3». Это значит, что для очень длинного вектора треть его элементов скорее всего будет равна 0, а остальные - 1. Наберите

» rand('seed',0); x=rand(100,l); % Внимание: х из (0,1)

» с=х>1/3

Здесь вторая строка предписывает выдать 0 для тех *i,* где *х*(*i*) «не проходит» проверку условия *х* > 1/3 и 1, если *х*(*i*) «проходит» ее. Предполагая, что элементы *х* равномерно распределены на [0,1], вы получите нули приблизительно в 1/3 случаев. Можете проверить это, выполнив *sum*(c)/100 и сравнив ответ с 2/3.

Однако можно ожидать большего, чем *с = х* > 1/3, выполнив

» rand('seed' ,0); x=rand(100,1); % Внимание: х из (0,1)

» c=round(x + 1/6)

Это связано с тем, что после сдвига на 1/6, только числа из *х*, меньшие 1/2 – 1/6 = 1/3, округляются до нуля. Нарисуйте картинку, иллюстрирующую эту идею. Подобным образом вы можете поступить с любой вероятностью *р.*

**Пример.** Взяв *seed* = 121, получим две случайные целочисленные матрицы размера 2×6 с элементами из [1,4] и [0,3] соответственно, после выполнения

» rand('seed',121); х = rand(2,6)

» у = ceil(4\*x) % Случай (1) с [1,4]

» z =floor(4\*x) % Случай (2) с [0,3]

Проверьте, что результатом будут *у* = 1 2 2 1 3 2

3 3 4 2 1 1

*z* = 0 1 1 0 2 1

2 2 3 1 0 0

## 3. Моделирование равномерных распределений

Мы обсудили, как функция *MATLAB* rand моделирует числа с равномерным распределением *U*(0,1). Кроме того, отмечалось, что можно использовать случайные числа, чтобы промоделировать равномерное распределение в любом интервале, сочетая *U*(0,l) с линейным преобразованием.

Напишите М-файл моделирующий 5000 бросаний трех игральных костей, с помощью функции *myunirand*(*а,b*), генерирующей равномерно распределенную случайную величину в диапазоне (*а,b*). Постройте гистограмму и подсчитайте число выпадений граней.

*Подсказка.* Изучите *help* функции *hist*().

## 4. Моделирование нормальных распределений

Чтобы получить столбец из 1000 случайных чисел с нормальным распределением, у которого среднее 0 и стандартное отклонение 1, наберите

» randn('seed',12); x=randn(1000);

где параметр seed схож по смыслу с используемым в *rand*, но учитывает особенности *randn*. Снова используя линейное преобразование, чтобы получить числа со средним *m* и стандартным отклонением *s*, наберите

» у = m + s\*x;

Напишите *М*-файл, моделирующий рост жителей в двух различных регионах, со средним *т =* 3, стандартным отклонением *s =* 5 и выборкой по *N* = 2000 человек в каждом из регионов, с помощью функции *mynormrand*(*m,s,N*). Постройте график.

И опять, как и с *rand*, получите семь столбцов из девяти случайных целых чисел с нормальным распределением, у которого среднее *m* = 3 и стандартное отклонение *s* = 4, сочетая r*andn* с любой из функций *ceil*, *fix*, *floor* и *round*.

*Замечание.* Под именами *rand* и *randn* известны два различных распределения. Однако команды *rand*('*normal*') и *rand*('*uniform*') заставят *rand* переключиться от одного распределения к другому!

## 5. Моделирование экспоненциальных распределений

Так как экспоненциальное распределение связано с равномерным, его легко промоделировать, используя генератор случайных чисел *MATLAB rand*. Чтобы промоделировать экспоненциальное распределение, у которого и *среднее,* и *стандартное отклонение* равны *l,* мы можем использовать следующее соотношение , предполагая, что *х* имеет равномерное распределение *U*(0,l).

Напишите *М*-файл, моделирующий экспоненциальное распределение с помощью функции *myexprand*(*l,N*). Сравните графики распределений (функция bar()) по выборкам из *N* = 20000 чисел и имеющих

* равномерное распределение на (0,2.4) со средним *m =* 1.2;
* нормальное распределение со средним *m =* 1.2 и стандартным отклонением *s* = 1.2;
* экспоненциальное распределение со средним *l* = 1.2.

Проверьте, что для последнего распределения и среднее, и стандартное отклонения оба равны *l = т,* что легко сделать с помощью команд *MATLAB* mean и std.

Времена между наступлениями событий обычно имеют экспоненциальное распределение. Промоделируйте вышеуказанным экспоненциальным распределением очередь из первых 50 клиентов к банкоматам в часы пик, если известно, что (в среднем) бывает приблизительно 131 клиентов между 9 часами утра и полуднем.

*Подсказка.* Каждый момент наступления события есть кумулятивная (накопленная) сумма времен между событиями. В *MATLAB* команда вычисления кумулятивной суммы:

» t = cumsum(z);

**ЗАДАНИЕ К ПРАКТИЧЕСКОМУ ЗАНЯТИЮ**

**Вариант 1**

По оценкам отдела торговли электротоварами ГУМ «Россия» в течение недели перед Рождеством электронные игры приобретают в среднем 140 покупателей в день с интервалами между покупками, распределенными по экспоненциальному закону. (Торговля происходит с 9 часов утра до 5 вечера.) В среднем 2/3 покупателей платят наличными, а остальные пользуются кредитными карточками; в 50% случаев покупают игровую приставку *Nintendo* и в 50% покупают *Sega*. Требуется промоделировать первые 30 покупок после открытия в предположении, что каждый посетитель делает покупку.

Напишите *М*-файл для определения времен между приходами и времен прихода первых 30 клиентов в день. Найдите среднее и стандартное отклонения времен между приходами.

Выведите таблицу первых 30 посещений в виде *Время посещения,* в минутах (один десятичный разряд после плавающей точки), *Способ оплаты* (наличные обозначим как +1, кредитной карточкой как –1) и *Купленный товар* (*Nintendo* как +1 и *Sega* как –1).

**Общая часть**

Установлено, что в среднем 15 поездов в час прибывают на станцию Новосибирск Главный с временами между прибытиями, имеющими экспоненциальное распределение. К тому же известно, что 75% составов являются местными электропоездами, число пассажиров в которых равномерно распределено на [0,80]. Остальные 25% являются междугородными поездами, число пассажиров в которых предполагается нормально распределенным со средним 90 и стандартным отклонением 20.

Определите времена прибытия первых 15 поездов и оцените общее число пассажиров, прибывающих в первые полчаса каждый день.

Постройте зависимости общего числа прибывших пассажиров как функции от времени прибытия для 5 дней на одном и том же графике.

**Вариант 2**

Напишите *М*-файл для построения 15×5-матрицы, содержащей типы (0 = междугородный, 1 = электричка) первых 15 поездов за 5 дней.

**Вариант 3**

Напишите *М*-файл для построения двух 15×5-матриц, содержащих оценку числа пассажиров на 15 электричках и 15 междугородных поездах в течение 5 дней.

**Вариант 4**

Напишите *М*-файл для построения двух 15×5-матриц, содержащих оценку общего числа пассажиров на первых 15 поездах в течение 5 дней.

*Подсказка.* Вспомните, как получить числа, принимающие значения 0 или 1 с вероятностями 1/3, 2/3, или каждое из них с вероятностью 1/2; как преобразовать вектор из 0 и 1 в вектор из –1 и +1.

*MATLAB* не может напечатать разные столбцы одной матрицы в разных форматах; они все должны быть целыми, либо все десятичными дробями. Чтобы избавиться от этого, нужно воспользоваться командами *fprintf* или *sprintf*. Например,

» а = pi, b = 35, с = 15

» с\_1 = [а b с]

» fprintf('с\_2 = %5.3f %5d %d\n' ,a,b,c)

» c\_3 = sprintf('%5.3f %5d %d\n' ,a,b,c)

где '%5.3*f*' означает печать вещественного числа пятью символами с тремя десятичными цифрами после плавающей точки, '%5*d*' означает печать целого из пяти цифр, тогда как '%*d*' выводит принятое по умолчанию число цифр.