Лабрадорная работа №2

Ярослава Галимова, M3236, вариант 2215

Задача 1

Пусть случайная величина X принимает ровно 2 значения: x_1 и x_2. Предположим противное. Пусть существуют такие A и B - невырожденные независимые случайные величины, что X = A + B. Так как они невырожденные, то они оба принимают хотя бы 2 различных значения (пусть для A это a_1, a_2, для B - b_1, b_2). Так как A и B независимы, то их совместная вероятность равна произведению вероятностей, а значит, любая пара (a_i, b_j) тоже имеет ненулевую вероятность и, более того, сумма $a_i + b_j = x_1$ или $a_i + b_j = x_2$.

Давайте теперь рассмотрим все попарные суммы. Не умаляя общности скажем, что $x_1 = a_1 + b_1$. Тогда, так как $a_1 \neq a_2$, то $x_2 = a_2 + b_1$. Аналогичными рассуждениями получаем следующие четыре выражения:

$$x_1 = a_1 + b_1$$
$$x_1 = a_2 + b_2$$
$$x_2 = a_1 + b_2$$
$$x_2 = a_2 + b_1$$

Тогда заметим, что $x_1 - x_2 = b_1 - b_2$ - следует из 1 и 3 выражений. А также $x_1 - x_2 = b_2 - b_1$ - следует из 2 и 4 выражений. Таким образом, $x_1 - x_2 = x_2 - x_1 \Rightarrow x_1 = x_2$, что противоречит тому, что X принимает 2 различных значения. Значит, мы доказали то, что от нас хотели.

Задача 2

R имеет распределение Рэлея. То есть:

$$f_R(r) = \frac{r}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}} \mathbb{1}(r \geq 0)$$

А также:

$$\theta \sim U[0, 2\pi] \Rightarrow f_\theta(x) = \frac{1}{2\pi}$$

Так как R и θ независимы по условию, то:

$$f(r, \theta) = f_R(r) * f_\theta(\theta) = \frac{r}{2\pi\sigma^2} e^{-\frac{x^2}{2\sigma^2}} \mathbb{1}(r \geq 0)$$

(На самом деле, по-хорошему тут можно было ещё домножить на характеристическую функцию от θ, так как по условию $\theta \sim U[0, 2\pi]$, но так получается, что в выражении θ не фигурирует, поэтому это не важно)
Выполним замену:

$$x = R\cos\theta$$
$$y = R\sin\theta$$

Это почти полярная замена (это обратная замена к полярной, так как мы из R и θ получаем x и y). Из курса любимого матанализа мы знаем, что это диффеоморфизм, а также мы знаем, что Якобиан такой замены равен R. А так как $JJ^{-1} = 1$, то $J^{-1} = \frac{1}{R}$. Отлично, давайте тогда проведём саму замену.

$$r = \sqrt{x^2 + y^2}$$
$$f_{X,Y} = f_{R,\theta}(\sqrt{x^2+y^2}, \theta) \cdot \frac{1}{\sqrt{x^2+y^2}}$$
$$f_{X,Y}(x,y) = \frac{\sqrt{x^2+y^2}}{2\pi\sigma^2} \cdot e^{\frac{-(x^2+y^2)}{2\sigma^2}} \cdot \frac{1}{\sqrt{x^2+y^2}} = (\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{x^2}{2\sigma^2}}) \cdot (\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{y^2}{2\sigma^2}})$$

Последнее равенство уже наталкивает нас на мысли о маргинальных распределениях, независимости X и Y, а также о том, что эти величины распределены нормально. Но давайте чуть более формально найдём $f_X(x)$ и $f_Y(y)$.

$$f_X(x) = \int_{-\infty}^{+\infty} f_{X,Y}(x,y)dy = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot \frac{1}{\sigma\sqrt{2\pi}} \cdot \int_{-\infty}^{+\infty} e^{\frac{-y^2}{2\sigma^2}} dy = \begin{bmatrix} t = \frac{y}{\sigma} \\ dt = \sigma dy \end{bmatrix} = \frac{1}{2\pi\sigma^2} \cdot e^{\frac{-x^2}{2\sigma^2}} \cdot \sigma\sqrt{2\pi}$$

В последнем равенстве мы сделали замену, чтобы выразить то, что мы хотим, через известный нам интеграл Пуассона. Приводя всё к нормальному виду и сокращая всё, что сокращается, получаем, что:

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{x^2}{2\sigma^2}}$$

Так как изначальная плотность $f_{X,Y}(x,y)$ была симметрична для x и y, то можно сразу сказать, что:

$$f_Y(y) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{y^2}{2\sigma^2}}$$

Таким образом оба распределения найдены. Они являются нормальными распределениями $N(0, \sigma^2)$, что и требовалось доказать в условии задания. Более того, мы сразу можем сказать, что $f_{X,Y} = f_X(x) \cdot f_Y(y)$, а значит, X и Y независимы.

Задание 3

$p(x) = 3x^2e^{-x^3} * \mathbb{1}(x \geq 0)$
Дисклеймер: весь код и результаты его запуска, невошедшие в этот файл с решением, лежат в подпапках с названиями вида $task\_3\_part\_n$ этого архива.

Пункт 1.

Для начала нужно было переопределить метод $\_pdf$, что я и сделала. А затем, замеряя время между началом генерации и её концом, я получила графики, характеризующие полученные значения. Это помогло мне убедиться в том, что всё написано корректно, так как график функции плотности распределения повторял полученную гистограмму. Давайте ради интереса взглянем на одну из них, полученную при генерации 50000 значений:

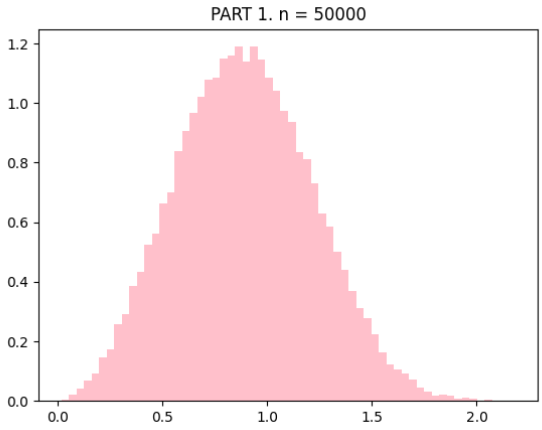

Рис. 1: rv_continuous

Я генерировала n = 100, 500, 1000, 10000 и 50000 точек, замеряя среднее время генерации одного числа. В данном случае среднее время генерации было примерно одинаковым для всех пяти значений n. Время генерации одного числа в среднем получилось примерно следующим:

n	Time (sec)	Average time (μs)
100	0.09396	939
500	0.48138	962
1000	0.93490	934
10000	9.36208	936
50000	46.88568	937

Таблица 1: Statistics

Из плюсов - очень просто написать. Из минусов - даже невооружённым взглядом видно, что генерируется всё довольно долго. Теперь давайте для сравнения напишем пункт 2 этого же задания.

Пункт 2.

В этом пункте нужно было найти обратную функцию к функции распределения (а значит, перед этим из плотности ещё и получить саму функцию распределения).

$$p(x) = 3x^2 e^{-x^3} \cdot \mathbb{1}(x \geq 0)$$
$$F(x) = \int_0^x p(t)dt$$
$$\int_0^x 3t^2 e^{-t^3} dt = \left[\begin{array}{c} u = t^3 \\ du = 3t^2 dt \end{array} \right] = \int_0^{x^3} e^{-u} du = 1 - e^{-x^3}$$
$$F(x) = 1 - e^{-x^3}$$
$$F^{-1}(y) = (-\ln(1-y))^{\frac{1}{3}}$$

Далее я немного видоизменила код из предыдущего пункта. Теперь за генерацию точек в соответствии с распределением отвечают следующие строки кода:

```
def F_inv(u):
    return (-np.log(1 - u))**(1/3)
```

А также эти:

```
samples = F_inv(np.random.uniform(size=n))
```

То есть точки генерируются с равномерным распределением, а затем к ним применяется обратная функция распределения, о чём и говорится в условии задания. Убедимся, что полученный результат отвечает изначальной плотности, рассмотрев гистограмму:

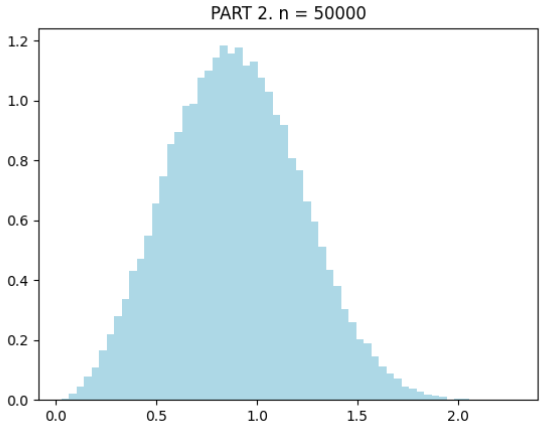

Рис. 2: F_inv

А также важно обратить внимание на замеры времени. Для этого снова покажу табличку:

n	Time (sec)	Average time (μs)
100	0.00005	0.47
500	0.00006	0.12
1000	0.00006	0.07
10000	0.00030	0.03
50000	0.00146	0.03

Таблица 2: Statistics

Теперь уже нам есть с чем сравнить замеры. Во-первых, для достаточно больших n второй метод работает в разы быстрее первого. Во-вторых, в отличие от первого метода, здесь мы получаем разное среднее время генерации одного числа. При $n = 100$ оно немного выше. Но если сначала сгенерировать 1000, а только потом 100 чисел, то среднее время генерации одного числа из 1000 тоже будет выше, так что, как мне кажется, это связано с внутренними оптимизациями языка, поэтому первая итерация цикла чуть более затратна по времени. В любом случае мы получили удовлетворяющий нас результат - если руками можно посчитать обратную к функции распределения, зная при этом только лишь плотность, то такой метод будет гораздо эффективнее.

Пункт 3. Выбранный метод - ratio of uniforms

Начнём с обоснования этого метода. В нём мы равномерно генерируем точки из множества $M = \{(u, v) \quad | \quad 0 \leq u \leq \sqrt{f(\frac{u}{v})}\}$. А раз генерируем равномерно, то плотность такого распределения будет равна $\frac{1}{S(M)}$, где S - мера (площадь) множества M.

$$p_{U,V}(u, v) = \frac{1}{S(M)} \cdot \mathbb{1}((u, v) \in M)$$

А затем делаем замену $x = \frac{u}{v}$. И введём ещё вторую "фиктивную"переменную (по ней мы в дальнейшем будем интегрировать для нахождения маргинального распределения, но давайте обо всё по порядку...). Назовём её $u_1 = u$.

$$\begin{bmatrix} x = \frac{v}{u} & \Rightarrow & v = xu_1 \\ u_1 = u & & u = u_1 \end{bmatrix}$$

$$J = \begin{vmatrix} 1 & 0 \\ x & u_1 \end{vmatrix} = u_1$$

$$p_{U_1,X}(u_1, x) = p_{U,V}(u_1, xu_1) \cdot |J| = p_{U,V}(u_1, v) \cdot u_1 = \frac{u_1}{S(M)} \cdot \mathbb{1}(0 \le u_1 \le \sqrt{f(x)}) =$$
$$\frac{u}{S(M)} \cdot \mathbb{1}(0 \le u \le \sqrt{f(x)})$$
$$p_X(x) = \int_0^{\sqrt{f(x)}} \frac{u}{S(M)} du = \frac{f(x)}{2 \cdot S(M)}$$

Из этого следует, что $p_X(x) = const \cdot f(x)$, то есть x имеет нужное распределение. Значит, мы доказали корректность метода.
Для написания этого метода требовалось найти границы генерации точек (u, v). В этом методе такими границами принято выбирать супремумы и инфинумы:

$$u \in [0, u\_max]$$
$$v \in [v\_min, v\_max]$$
$$u\_max = sup_x \sqrt{f(x)}$$
$$v\_min = inf_x x\sqrt{f(x)}$$
$$v\_max = sup_x x\sqrt{f(x)}$$

Это логично, поскольку мы хотим как можно более оптимально генерировать точки, то есть чтобы среди них неподходящих нам было мало. И при этом слишком сильно ограничивать область тоже нельзя, так как тогда мы лишимся корректности алгоритма. Поэтому нас интересуют супремумы и инфинумы. Их можно посчитать, взяв производную у функций, тут ничего интересного. В итоге мы получаем следующий код для генерации:

```
def ratio_of_uniforms(n):
    samples = []
    u_max, v_min, v_max = 0.53, 0, 0.65
    while len(samples) < n:
        u = uniform.rvs(scale = u_max)
        v = uniform.rvs(loc = v_min, scale = v_max-v_min)
        x_c = v / u
        if u <= x_c * np.exp(-x_c**3 / 2):
            samples.append(x_c)
    return np.array(samples)
```

Рассмотрим полученный график. Он действительно похож на то, что мы хотели:

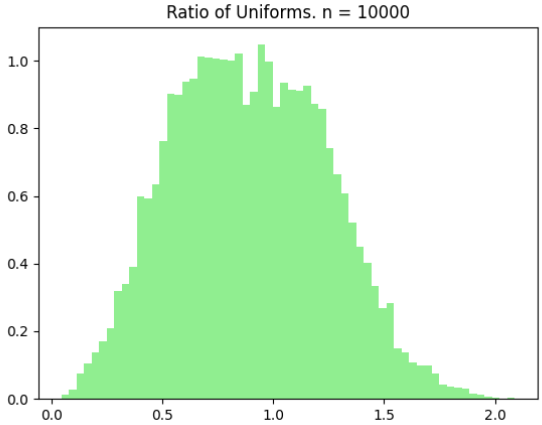

Рис. 3: Ratio of uniforms

А также обратим внимание на таблицу с замерами времени:

n	Time (sec)	Average time (μs)
100	0.00999	99.87
500	0.03428	68.56
1000	0.06163	61.63
10000	0.64867	64.87
50000	3.33965	66.79

Таблица 3: Statistics

Делаем выводы. Этот метод мне показался сильно сложнее в реализации, чем первые два. Но, наверное, это связано ещё с тем, что я пыталась разобраться в его корректности... В любом случае он требует расчёта границ для генерирования точек. С другой стороны, иногда такая задача может быть проще, чем расчёт обратной функции к функции распределения (но не в нашем случае, у нас в задании функция довольно простая). Метод примерно в 1000 раз более затратен, чем второй, зато всё ещё гораздо быстрее первого.

Задание 4

$f(x) = \lambda e^{-\lambda x}$
$x \geq 0$, так как это экспоненциальное распределение. Перед началом решения давайте посчитаем матожидание и дисперсию этого распределения.

$$E(x) = \int\limits_0^{+\infty} x\lambda e^{-\lambda x} dx = -xe^{-\lambda x}\Big|_0^{+\infty} + \int\limits_0^{+\infty} e^{-\lambda x} dx = \frac{1}{\lambda}$$

$$E(x^2) = \int\limits_0^{+\infty} x^2\lambda e^{-\lambda x} dx = -x^2 e^{-\lambda x}\Big|_0^{+\infty} + \int\limits_0^{+\infty} 2xe^{-\lambda x} dx = \frac{2}{\lambda^2}$$

$$D(x) = E(x^2) - E^2(x) = \frac{2}{\lambda^2} - (\frac{1}{\lambda})^2 = \frac{1}{\lambda^2}$$

Так как матожидание линейно, то:

$$E(\overline{X_n}) = \frac{1}{n} \cdot \sum_1^n E(X_i) = \frac{1}{n} \cdot n \cdot \lambda = \lambda$$

Так как события независимы, то:

$$D(\overline{X_n}) = \frac{1}{n^2} \cdot \sum_1^n D(X_i) = \frac{1}{n^2} \cdot n \cdot \frac{1}{\lambda} = \frac{1}{n\lambda^2}$$

Применим неравенство Чебышева:

$$P(|\overline{X_n} - \tfrac{1}{\lambda}| \geq \varepsilon) \leq \tfrac{1}{n\lambda^2\varepsilon^2} \quad \Rightarrow \quad P(|\overline{X_n} - \tfrac{1}{\lambda}| \leq \varepsilon) \geq 1 - \tfrac{1}{n\lambda^2\varepsilon^2} \quad \Rightarrow \quad \tfrac{1}{n\lambda^2\varepsilon^2} \leq \delta \quad \Rightarrow \quad n \geq \tfrac{1}{\lambda^2\delta\varepsilon^2}$$

Ответ: $n_{\varepsilon,\delta} = \lceil \frac{1}{\lambda^2\varepsilon^2\delta} \rceil$

Теперь нужно доказать через ЦПТ.
Для достаточно больших n по ЦПТ известно, что:

$$\overline{X_n} \approx N(E(x); \frac{D(x)}{n})$$
$$\overline{X_n} \approx N(\frac{1}{\lambda}; \frac{1}{\lambda^2 n})$$
$$\frac{\overline{X}_n - \frac{1}{\lambda^2}}{\sqrt{\frac{1}{\lambda^2 n}}} \xrightarrow[n\to\infty]{} N(0,1)$$

Мы хотим, чтобы $P(|\overline{X_n} - E(x)| \le \varepsilon) \ge 1 - \delta$

То есть это можно записать следующим образом $P\left(|\frac{\overline{X_n}-E(x)}{\sqrt{D(x)/n}}| \le \frac{\varepsilon}{\sqrt{D(x)/n}}\right) \ge 1 - \delta$

Пусть F - функция нормального распределения. Она имеет вид $F(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \int\limits_{-\infty}^{x} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx$.

Это, конечно, пугает, но в дальнейшем нам нужно будет только посчитать значение обратной к этой функции в нескольких точках (то есть квантили). Давайте выразим то, что у нас есть, через F:

$$P\left(\left|\frac{\overline{X_n}-E(x)}{\sqrt{D(x)/n}}\right| \le \frac{\varepsilon}{\sqrt{D(x)/n}}\right) \ge 1 - \delta$$
$$P\left(-\frac{\varepsilon}{\sqrt{D(x)/n}} \le \frac{\overline{X_n}-E(x)}{\sqrt{D(x)/n}} \le \frac{\varepsilon}{\sqrt{D(x)/n}}\right) \ge 1 - \delta$$
$$P\left(\left|\frac{\overline{X_n}-E(x)}{\sqrt{D(x)/n}}\right| \le \frac{\varepsilon}{\sqrt{D(x)/n}}\right) = 2F\left(\frac{\varepsilon}{\sqrt{D(x)/n}}\right) - 1 \ge 1 - \delta$$

Последнее равенство выполняется из-за симметричности графика нормального распределения относительно вертикальной прямой, проведённой из точки E(x), а в нашем случае распределение N(0, 1), то есть симметрия будет относительно нуля. То есть $P(|A| \le a) = P(A \le a) - P(A \le -a) = F(a) - F(-a)$, а $F(-a) + F(a)$ образуют всё множество, то есть эта сумма равна 1. Тогда $F(-a) = 1 - F(a)$.

$$2F\left(\frac{\varepsilon}{\sqrt{D(x)/n}}\right) - 1 \ge 1 - \delta$$
$$F\left(\frac{\varepsilon}{\sqrt{D(x)/n}}\right) \ge \frac{2-\delta}{2}$$
$$\frac{\varepsilon}{\sqrt{D(x)/n}} \ge F^{-1}(1 - \frac{\delta}{2})$$
$$\varepsilon\sqrt{n} \ge F^{-1}(1 - \frac{\delta}{2})\sqrt{D(x)}$$
$$n \ge \frac{F^{-1}(1-\frac{\delta}{2})\sqrt{D(x)}}{\varepsilon}$$
$$n \ge \left(\frac{F^{-1}(1-\frac{\delta}{2})\sqrt{D(x)}}{\varepsilon}\right)^2$$

Подставив всё, что нам известно, получим ответ: $n_{\varepsilon,\delta} = \lceil\left(\frac{F^{-1}(1-\frac{\delta}{2})}{\varepsilon\lambda}\right)^2\rceil$

В обеих полученных формулах параметр λ находится в знаменателе функции, а значит, чтобы n удовлетворяло неравенству при любой λ, нужно взять $\lambda = 1$, так как при таком значениии параметра значение функции максимально. Посчитаем n в обоих случаях:

$$n_{\varepsilon,\delta} = \lceil\frac{1}{\lambda^2\varepsilon^2\delta}\rceil = \lceil\frac{1}{(0.01)^2 \cdot 0.05}\rceil = 200000$$
$$n_{\varepsilon,\delta} = \lceil\left(\frac{F^{-1}(1-\frac{\delta}{2})}{\varepsilon\lambda}\right)^2\rceil = \lceil(\frac{F^{-1}(0.975)}{0.01})^2\rceil = (1.96/0.01)^2 = 196^2 = 38416$$

Казалось бы, вторым способом мы получили гораздо более точный результат, но чего нам это стоило...
Давайте посмотрим, как это работает на практике, написав код на python'е.

```
counter_1 = 0
for _ in range(100):
    sample = np.random.exponential(scale = 1, size=n_1)
    mean = np.mean(sample)
    if abs(mean - 1) <= epsilon:
        counter_1 += 1
```

Это код (который также можно посмотреть в отправленном архиве в папке task_4), генерирующий точки с заданным распределением и проверяющий точки на соответствие неравенству. После его запуска получаем следующий вывод:

```
Неравенство Чебышёва в процентах (n = 200000): 1.00
ЦПТ  в процентах (n = 38416): 0.97
```

Осталось только лишь сделать вывод. Неравенство Чебышева, конечно, показалось мне гораздо более простым. Но выдало оно огромный результат - аж 200000. Зато в итоге мы получили 100% результат. ЦПТ же в самом начале заменяет функцию на эквивалентную. А эквивалетны они при $n \to +\infty$, то есть даже для достаточно больших n мы всё равно будет допускать некоторую неточность. Тем не менее, результат оказался тоже вполне хорошим - 97%. На этом всё!

P.S. - Так как работа лабрадорная, но лабрадора у меня нет, пусть тут будет самое похожее на лабрадора, что у меня имеется:

Рис. 4: Моя супер-мега-кругая собака Ева