Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 24.М41-мм

Обзор и сравнение инструментов для работы с вероятностными распределениями

Михайлов Михаил Дмитриевич

Отчёт по учебной практике
в форме «Сравнение»

Научный руководитель:
ст. преподаватель кафедры системного программирования, к. ф.-м. н., Гориховский В. И.

Санкт-Петербург
2025

Оглавление

Введение **3**

Цели и задачи работы **4**

Обзор **5**

Обзор предметной области 5

Обзор литературы 21

Обзор решений 27

Выводы 36

1. Сравнение инструментов **38**

1.1. Сравнение функциональности 38

1.2. Сравнение особенностей реализации 41

1.3. Выводы 44

2. Требования к ядру PySATL **46**

2.1. Диаграмма вариантов использования 46

2.2. Функциональные требования 47

Заключение **50**

Список литературы **52**

Введение

Разработка ПО и алгоритмических подходов для решения задач статистики и стохастического моделирования ведется с 1940-х годов, когда появились первые алгоритмы для генерации случайных величин [49], [69]. В 1960 годах начали появляться первые программные решения для исполнения статистических процедур, в частности для проверки гипотез и оценки параметров (подробнее см. [72]).

С течением времени, появилось множество программных решений для задач статистики и еще большее число теоретических методов. Это привело к тому что пользователям многих решений просто недоступны современные методы статистики [70]. На сервисе публикации препринтов arXiv[1] количество материалов опубликованых в архиве `stat` (статистика), за последний год, составляет более 15000, среди которых только 875 содержит тег `stat.CO` (статистические вычисления, визуализация), что свидетельствует о том, что многие теоретические методы скорее всего не реализованы ни в одном существующем решении.

Проект PySATL [58] ставит своей целью предоставить единообразный доступ как классическим, так и современным методам математической статистики. На данный момент в рамках проекта возникла необходимость в разработке общего ядра. Сейчас в качестве ядра выступает библиотека SciPy [68], однако разработчики проекта сообщают что функциональности этой библиотеки не хватает и в некоторых местах библиотека не обладает достаточной гибкостью. Поэтому, прежде чем разрабатывать, общее ядро необходимо понять есть ли решения, кроме SciPy, на основе которых можно было бы базировать разработку.

В настоящей работе рассмотрены математические и алгоритмические аспекты работы с вероятностными распределениями. Выбраны несколько библиотек, которые могут претендовать на роль общего ядра и произведен анализ их функциональности и особенностей реализации. На основании этого сформулированы рекомендации к разработке прототипа и функциональные требования к ядру.

[1] https://arxiv.org (дата обращения: 9 января 2025 г.).

Цели и задачи работы

Общей целью работы является разработка и внедрения общего ядра для проекта PySATL.

Целью работы на этот семестр является сравнительный анализ возможностей, производительности и корректности различных инструментов для работы с вероятностными распределениями. Работа фокусируется на задачах связанных с вычислением числовых характеристик/-функционалов и моделированием распределений — задачи статистического вывода не рассматриваются. Выделяются следующие задачи:

1. Выделить основные инструменты для работы с вероятностными распределениями и сравнить их возможности.
2. Разработка методики сравнения сэмплирования.
3. Реализация бенчамарк-системы для выделенных инструментов и проведение сравнительного анализа корректности, точности и производительности инструментов.
4. Формулировка требований к ядру проекта PySATL.

Обзор

В этом разделе представлен обзор актуальных задач, возникающих при работе с распределениями; литературы, посвященной методам решения этих задач и анализу качества получающихся решений; существующего ПО для решения этих задач. Этот раздел разделен на три части.

- В разделе «обзор предметной области» даны основные сведения о вероятностных распределениях и их математических характеристиках, а также прикладных задачах в которых они возникают;
- В разделе «обзор литературы» рассмотрены основные научные источники посвященные алгоритмическим аспектам при работе с вероятностными распределениями;
- В разделе «обзор существующих решений» дается представление о решениях, рассматриваемых в работе.

Обзор предметной области

Основным понятиям в статистике и стохастическом моделировании является *распределения случайной величины* или, более общо, *распределения случайного объекта* (далее, под случайной величиной понимается любой случайный объект, реализации которого необязательно суть вещественные числа) [36]. Ниже изложены основные теоретические сведения касающиеся случайных величин, а также задач в которых они возникают, в соответствии с монографиями [78] и [80].

Случайные величины и способы их задания

Для случайной величины ξ, принимающей значения в некотором пространстве $\mathcal{X}$, её распределением называется (см. [80]) вероятностная мера $\mathbb{P}_\xi(\cdot)$ на $\mathcal{X}$, такая что $\mathbb{P}_\xi(A)$ есть вероятность того что реализация ξ попадет в множество $A \subset \mathcal{X}$[2].

[2]Строго говоря, $\mathcal{X}$ должно быть снабжено некоторой σ-алгеброй $\mathcal{F}$, и $\mathbb{P}_\xi$ должна быть определена только для $A \in \mathcal{F}$. Иначе говоря, тройка $(\mathcal{X}, \mathcal{F}, \mathbb{P}_\xi)$ должна образовывать вероятностное пространство.

Как правило (см. например [78]), выделяют следующие виды случайных величин

- *Дискретные случайные величины.*

  В этом случае $\mathcal{X}$ представляет собой некоторое дискретное (конечное или счетное) множество. Например, число выпадений монеты орлом при нескольких бросках (биномиальное распределение); уровень образования у случайно выбранного человека (категориальное распределение); случайная величина которая принимает одно значение (вырожденное распределение);

- *Одномерные непрерывные случайные величины*[3].

  В этом случае $\mathcal{X} = \mathbb{R}$ или $\mathcal{X} \subset \mathbb{R}$ ненулевой меры. Такие величины используются для описания случайных времен, расстояний и т.д. Согласно [36] важными представителями являются: равномерное распределение $\mathcal{U}(a; b)$, нормальное распределение $\mathcal{N}(\mu, \sigma^2)$ и экспоненциальное распределение $\mathrm{Exp}(\lambda)$;

- *Многомерные непрерывные случайные величины.*

  В этом случае $\mathcal{X} \subseteq \mathbb{R}^d$, ненулевой меры. Во много многомерные случайные величины являются аналогами одномерных непрерывных случайных величин, однако решение стандартных задач, таких как моделирование или вычисление числовых характеристик затруднено из-за проклятия размерности [14].

Отдельное направление статистики работает с данными о направлении, в связи с этим часто можно также отдельно выделить следующую категорию случайных величин.

- *Случайные геометрические примитивы.*

  Примерами таких случайных величин служат случайные углы или случайные матрицы симметрий. Согласно [56], геометрической случайной величиной называется случайная величина принимающая значения на замкнутой и ограниченной поверхности в евклидовом пространстве.

[3]Здесь и далее под непрерывными случайными величинами подразумеваются абсолютно непрерывные случайные величины, т.е. распределение которых имеет плотность относительно меры Лебега

С точки зрения ПО, работа с распределением, как с вероятностной мерой, является неудобной, так как компьютер не может работать с произвольными множествами. Однако, как правило, с распределением можно связать некоторую функцию, которая полностью определяет распределение. Так, чтобы идентифицировать распределение дискретной случайной величины, достаточно знать *функцию вероятности*, определяемую равенством (pmf).

$$f_\xi(x) = \mathbb{P}_\xi(\{x\}),\ \text{т.е. вероятность того что } \xi = x,\ x \in \mathcal{X} \qquad \text{(pmf)}$$

Если на пространстве возможных значений $\mathcal{X}$ задана некоторая мера μ, плотностью распределения $\mathbb{P}_\xi$ относительно μ называется[4] такая функция $f_\xi(x)\colon \mathcal{X} \to \mathbb{R}$, что выполнено тождество (pdf):

$$\mathbb{P}_\xi(A) = \int_A f_\xi(x)\, d\mu \qquad \text{(pdf)}$$

В случае если μ это считающая мера, плотность f_ξ определяется равенством (pmf), в случае если μ это мера Лебега, говорят просто о плотности непрерывной случайной величины/случайного вектора.

Несмотря на то, что плотность распределения полностью его характеризует, для того чтобы вычислять вероятности $\mathbb{P}_\xi(A)$, необходимо производить интегрирование (или суммирование), поэтому для некоторых задач представление распределения в виде плотности является неудобным. В частности, если ξ — случайная величина (дискретная или непрерывная) принимающие значения из $\mathbb{R}^d$, довольно часто приходится смотреть на вероятность попадания в некоторую ячейку $\langle \mathbf{a}; \mathbf{b} \rangle$. Под ячейкой подразумевается множество:

$$\langle \mathbf{a}; \mathbf{b} \rangle = \left\{ \begin{pmatrix} c_1 \\ \vdots \\ c_d \end{pmatrix} \in \mathbb{R}^d \;\middle|\; a_1 < c_1 \leqslant b_1, \cdots, a_d < c_d \leqslant b_d \right\}$$

В случае если $a = -\infty$ или $b = +\infty$ подразумевается соответствующая бесконечная ячейка.

[4]Условия существования плотности описываются теоремой Радона-Никодима, см. например [41]

Для доступа к вероятностям попадания в ячейку эффективнее работать с *функцией распределения случайной величины*, определяемой равенством (cdf).

$$F_\xi(\mathbf{x}) = \mathbb{P}_\xi(\langle -\infty; \mathbf{x}\rangle) \tag{cdf}$$

В этом случае $\mathbb{P}_\xi(\langle \mathbf{a}; \mathbf{b}\rangle)$ выражается через значения $F_\xi(\cdot)$ с помощью формулы включения-исключения [78].

С понятием функции распределения тесно связано понятие *квантильной функции*. Для случайной величины ξ со значениями из $\mathbb{R}$, её квантильная функция определяется равенством (ppf) (подробно о различных определениях см. в обзоре [40]).

$$\omega_\xi(p) = \inf_u \{F_\xi(u) \geqslant p\} \tag{ppf}$$

Известно что, распределения случайных величин $\omega_\xi(U)$, где $U \sim \mathcal{U}[0; 1]$, и ξ совпадают. В случае, когда $F_\xi(\cdot)$ строго возрастает на всей области определения, квантильная функция является обратной функцией в обычном смысле $\omega_\xi(\cdot) = F_\xi^{-1}(\cdot)$. Отдельно следует отметить что существуют обобщения понятия квантильной функции на случай случайных величин со значениями из $\mathbb{R}^d$ [23], однако для их вычисления необходимо решать уравнения в частных производных [20].

Существуют и другие функциональные характеристики распределения, многие из которых приходят из анализа выживаемости [34]. *Функцией выживаемости* называется функция определяемая равенством (sdf).

$$S_\xi(\mathbf{x}) = 1 - F_\xi(\mathbf{x}) \tag{sdf}$$

Для случайных величин со значениями из $\mathbb{R}$, функцией интенсивности отказов и кумулятивной функцией интенсивности отказов называются функции определяемые равенствами (hrdf) и (chdf) соответственно.

$$h_\xi(x) = -\frac{S'_\xi(x)}{S_\xi(x)}; \tag{hrdf}$$

$$H_\xi(x) = -\ln(S_\xi(x)); \tag{chdf}$$

Однако, некоторые распределения, например α-устойчивые распре-

деления [78], не допускают явного задания с помощью плотности или функции распределения, однако допускают задания с помощью так называемых *интегральных преобразований*. Такие распределения все чаще возникают в современных моделях стохастического анализа, (см. например [6]). В случае случайной величины ξ со значениями из $\mathbb{R}$, её

- *Характеристической функцией* называется преобразование Фурье, определяемое равенством (cf);

$$\varphi_\xi(u) = \int_{\mathbb{R}} \exp(itu)\mathbb{P}_\xi(dt), \quad u \in \mathbb{R} \tag{cf}$$

- *Момент-производящей функцией*[5] называется преобразование, задаваемое равенством (mgf).

$$M_\xi(u) = \int_{\mathbb{R}} \exp(tu)\mathbb{P}_\xi(dt), \quad u \in \mathbb{R} \tag{mgf}$$

Характеристическая функция всегда существует и полностью определяет распределение случайной величины [80]. В свою очередь, момент-производящая функция существует не всегда, но в тех случаях когда существует, также однозначно характеризует распределение. В случае когда ξ принимает только неотрицательные значения, определены также *преобразование Лапласа* и *преобразование Меллина*, задаваемые равенствами (lt) и (mt) соответственно.

$$\mathcal{L}_\xi(u) = \int_{\mathbb{R}_+} \exp(-tu)\mathbb{P}_\xi(dt), \quad u \in \mathbb{R}_+ \tag{lt}$$

$$\mathcal{M}_\xi(u) = \int_{\mathbb{R}_+} t^u\mathbb{P}_\xi(dt), \quad u \in \mathbb{R}_+ \tag{mt}$$

Эти преобразования однозначно также характеризуют распределение случайной величины [78], [21]. В случае если ξ многомерная случайная величина, определяется характеристическая функция (см. [80]), преобразования (mgf), (lt), (mt) в некоторых ситуациях допускают обобщение на многомерный случай, см. например [2].

[5]Также называется производящей функцией моментов

На рис. 1 схематично изображены основные способы задания непрерывных вероятностных распределений, и связь между ними. Так как в теории многомерных квантилей нет результатов напрямую выражающих квантильные функции через плотности или интегральные преобразования, переходы которые имеют место только в одномерном случае, изображены пунктирными стрелками.

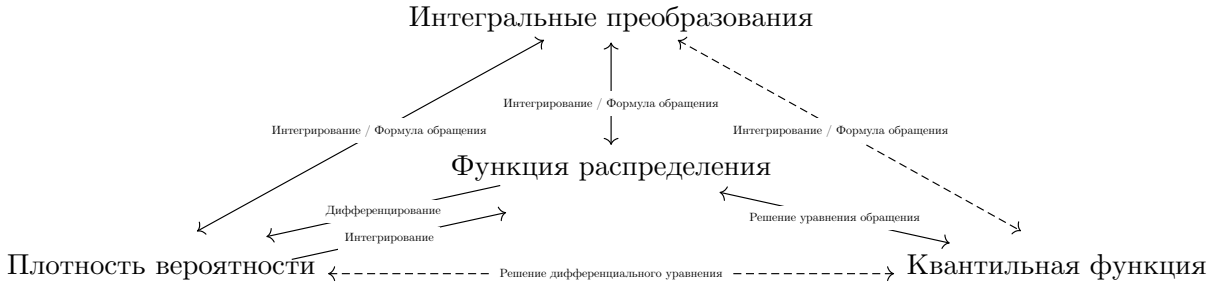

Рис. 1: Способы задания непрерывных распределений

Замечение. Плотность и функция распределения непрерывной случайной величины со значениями в $\mathbb{R}^d$ связаны соотношениями

$$f_\xi(\mathbf{x}) = \frac{\partial F_\xi}{\partial x_1 \cdots \partial x_d}(\mathbf{x}) \quad F_\xi(\mathbf{x}) = \int\limits_{\langle -\infty;\mathbf{x}\rangle} f_\xi(\mathbf{t})\, d\mathbf{t} \tag{1}$$

Формулы обращения для интегральных преобразований представлены в [80], [21]. Отдельно стоит отметить, что в работе [64] показано как можно вычислять квантильную функцию по плотности распределения и наоборот, не прибегая к вычислению функции распределения. Этот подход может оказаться полезным при работе с достаточно сложными плотностями.

Семейства вероятностных распределений

В задачах статистики, как правило, оперируют не с одним каким-то конкретным распределением, а с набором распределений, из которого надо выбрать наиболее подходящее, или проверить какую-то гипотезу. Более строго, *параметрическим семейством распределений* называется некоторое множество $\{\mathbb{P}_\theta\}_{\theta\in\Theta}$ распределений, зависящих от скалярного

или векторного параметра θ, Θ — множество возможных значений параметра [79].

Для любого распределения $\mathbb{P}_\xi$ случайной величины ξ определено семейство локации и масштаба, т.е. семейство распределений всех аффинных преобразований величины ξ:

$$\mathrm{loc} + \mathrm{scale} \cdot \xi \sim \mathbb{P}^{\xi}_{(\mathrm{loc},\mathrm{scale});} \qquad \text{(loc-scale-family)}$$

где параметры $\mathrm{loc}, \mathrm{scale} \in \mathbb{R}$ для вещественнозначных случайных величин, и $\mathrm{loc} \in \mathbb{R}^d$, $\mathrm{scale} \in \mathbb{R}^{d\times d}$ для векторозначных случайных величин. Примером такого семейства является семейство нормальных распределений $\mathcal{N}(\mu, \sigma)$, определяемых равенством (normal-family).

$$\mu + \sigma \cdot \xi, \quad \xi \sim \mathcal{N}(0,1), \text{ т.е. } f_\xi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \qquad \text{(normal-family)}$$

Более общим понятием является понятие *семейства замкнутого относительно действия группы*, см. [43] и [55].

Другим, в некотором смысле ортогональным, понятием является понятие *экспоненциального семейства распределений* [5]. Параметрическое семейство распределений $\{\mathbb{P}_\theta\}_{\theta\in\Theta}$ относится к экспоненциальному типу, если плотности (или функции вероятностей) которых можно записать в виде (exp-family) [6].

$$f(\mathbf{x}|\theta) = \exp(\langle \mathbf{T}(\mathbf{x}), \vec{\eta}(\theta)\rangle + A(\mathbf{x}) + D(\theta)) \qquad \text{(exp-family)}$$

Многие распространенные семейства распределений являются экспоненциальными, см. [53]. Для моделей относящихся к экспоненциальным семействам существует богатая теория оценивания параметров [43]. Большой список параметрических семейств и связывающие их соотношения представлены в [42].

Приведенные выше семейства интересны с точки зрения теоретической статистики. С точки зрения прикладной статистики, интерес представляют распределения, которые допускают гибкость в плане оцени-

[6] При этом требуется чтобы множество точек $\mathbf{x}$, в которых плотность отлична от 0, не зависело от параметра θ

вания параметров: так, для нормального распределения два параметра определяют не только его локацию и масштаб, но и всю форму распределения, причем такое поведение присуще не только нормальному распределению. Для того чтобы решить эту проблему, было предложено несколько «гибких» семейств распределений, среди которых широко распространены семейство распределений Пирсона [17] и металогическое семейство [32].

Отдельно стоит отметить, что многие параметрические семейства зачастую имеют несколько параметризаций, каждая из которых может быть удобна в том или ином контексте, например в работе [57] приведены четыре параметризации для обобщенного гиперболического распределения. Множество других различных параметризаций для одних и тех же семейств собраны в базе проекта ProbOnto [65].

Преобразования случайных величин

Во многих моделях распределения могут быть составлены из более «простых» распределений с помощью различных методов. В [5] отмечается что, в контексте статистического вывода, любая модель для данных может рассматриваться как вероятностное распределение. Существует множество способов комбинировать и преобразовывать вероятностные распределения, интересная практическая реализация этого взгляда доступна в библиотеке Pomegranate, [61]. Ниже рассмотрены основные способы для непрерывных вещественнозначных случайных величин, большинство которых относят к теории алгебры случайных величин, см. [63].

- *Аффинное преобразование.*

  Если случайная величина ξ имеет распределение с функцией плотности $f_\xi(x)$, то плотность её линейного преобразования $a\xi + b$ описывается равенством (aff-tr);

$$f_{a+b\xi}(y) = \frac{1}{|a|} f_\xi\left(\frac{y-b}{a}\right), \quad a \neq 0. \tag{aff-tr}$$

- *Биективное преобразование.*

  Плотность распределения случайной величины $g(\xi)$, в случае строго монотонной функции g, определяется с помощью равенства (bij-tr).

$$f_{g(\xi)}(y) = f_\xi(g^{-1}(y)) \cdot \left| \frac{d}{dy} g^{-1}(y) \right|. \qquad \text{(bij-tr)}$$

  Уравнение (aff-tr) является частным случаем (bij-tr). Для немонотонных функций распределение вычисляется с использованием разбиения на участки монотонности;

- *Распределение суммы независимых случайных величин.*

  Если случайные величины ξ_1 и ξ_2 независимы[7], то плотность суммы $\xi_1 + \xi_2$ вычисляется с помощью свёртки (sum-rv);

$$(f_{\xi_1} \oplus f_{\xi_2})(z) = \int_{\mathbb{R}} f_{\xi_1}(x) f_{\xi_2}(z - x)\, dx. \qquad \text{(sum-rv)}$$

- *Распределение произведения независимых случайных величин.*

  Для независимых случайных величин ξ_1 и ξ_2, плотность их произведения $\xi_1 \cdot \xi_2$ задается с помощью мультипликативной свертки (prod-rv).

$$(f_{\xi_1} \odot f_{\xi_2})(z) = \int_{\mathbb{R}} \frac{1}{|x|} f_{\xi_1}(x) f_{\xi_2}\left(\frac{z}{x}\right) dx. \qquad \text{(prod-rv)}$$

Отдельно стоит отметить что важную роль играют *порядковые статистики* (см. [44]). Если случайный вектор $\xi = (\xi_1, \xi_2, \ldots, \xi_d)$ имеет независимые и одинаково распределённые с плотностью распределения $f(x)$ и функцией распределения $F(x)$ компоненты, то их порядковые статистики $\xi_{(1:d)} \leqslant \xi_{(2:d)} \leqslant \ldots \leqslant \xi_{(n)}$ имеют плотности, описываемые равенством (ord-stat);

$$f_{\xi_{(k)}}(x) = \frac{n!}{(k-1)!(n-k)!} [F(x)]^{k-1} [1 - F(x)]^{n-k} f(x). \qquad \text{(ord-stat)}$$

[7] Т.е. $\mathbb{P}(\xi_1 \in B_1 \text{ и } \xi_2 \in B_2) = \mathbb{P}(\xi_1 \in B_1) \cdot \mathbb{P}(\xi_2 \in B_2)$ верно для всех $B_1, B_2 \subset \mathbb{R}$, являющихся борелевскими

Отметим, что с помощью численных методов, возможно вычисление распределений порядковых статистик и для векторов с независимыми, но необязательно одинаково распределенными компонентами.

Другие две важные операции которые возникают при работе со случайными векторами — маргинализация (взятие проекции) и вычисление порядковых статистик.

- *Проекции.*

  Для случайного вектора $\xi = (\xi_1, \ldots, \xi_d)$ (с возможно зависимыми компонентами) с функцией распределения $F_\xi(\mathbf{x})$, функция распределения случайного «подвектора» $(\xi_{i_1}, \ldots, \xi_{i_k})$, описывается пределом (proj-tr);

$$F_{\mathrm{Pr}(\xi;i_1,\ldots,i_k)}(x_{i_1}, \ldots, x_{i_k}) = \lim_{\substack{x_i \to +\infty \\ i \neq i_1,\ldots,i_k}} F_\xi(x_1, \ldots, x_d) \qquad \text{(proj-tr)}$$

- *Длины и углы.*

  Как правило, под случайным вектором понимается случайный элемент $\mathbb{R}^d$ представимый своими координатами. Однако, в некоторых ситуациях (см. например [24]) куда удобнее оперировать со сферическими или другими координатами. В этом случае работает многомерный аналог формулы (bij-tr).

В приложениях часто возникают понятия *цензурированых* и *урезанных* распределений [19]. Для распределения $\mathbb{P}_\xi$ случайной величины ξ принимающей вещественные значения, урезанным называется условное распределение, определяемое равенством (truncated-dist).

$$\mathbb{P}_{\mathrm{Truncated}(\xi,L,R)}(B) = \frac{\mathbb{P}_\xi([L;R] \cap B)}{\mathbb{P}_\xi([L;R])} \qquad \text{(truncated-dist)}$$

Распределение (truncated-dist) это условное распределение ξ если априори известно, что значение ξ лежит в отрезке $[L;R]$. В принципе, это понятие является просто частным случаем условного распределения случайной величины.

В свою очередь, цензурированым на отрезке $[L; R]$ распределением называется распределение случайной величины, определяемой равенством (censored-dist).

$$\mathrm{Censored}(\xi, L, R) = \begin{cases} L & \xi < L \\ \xi & L \leqslant \xi \leqslant R \\ R & R < \xi \end{cases} \qquad \text{(censored-dist)}$$

Такие распределения часто возникают в задачах регрессионного анализа и при работе с эконометрическими данными см. [8].

Помимо трансформации одного распределения в другое и алгебраических операций над распределениями, еще одним способом образования «сложного» распределения из нескольких простых является образование *смесей*.

- *Дискретная смесь.*

  Под дискретной смесью подразумевают комбинацию конечного числа распределений, каждое из которых взвешено определённым коэффициентом. Функция распределения дискретной смеси случайных величин $\xi_1, \ldots, \xi_n$ с весами $\mathbf{w} = (w_1, \ldots, w_n)$ задаётся равенством (dmix);

$$F_{\mathrm{mix}(\mathbf{w};\xi)}(x) = \sum_{i=1}^{n} w_i F_{\xi_i}(x), \quad \sum_{i=1}^{n} w_i = 1 \qquad \text{(dmix)}$$

- *Непрерывная смесь.*

  Является непрерывным аналогом дискретной смеси. Пусть $F(x \,|\, \theta)$ семейство плотностей, зависящее от параметра $\theta \in \Theta \subset \mathbb{R}^n$. Если на Θ задано некоторое распределение параметров с плотностью $\omega(\theta)$, непрерывная смесь плотностей $F(x \,|\, \theta)$ определяется равенством (cmix).

$$F_{\mathrm{mix}(\omega;F_\xi(\cdot|\theta))}(x) = \int_\Theta F_\xi(x \mid \theta) \cdot w(\theta)\, d\theta \qquad \text{(cmix)}$$

Смеси широко применяются в кластерном анализе и для изучения ядерных оценок плотности. Общая теория смесей в абстрактном случае и конкретные примеры приведены в работе [10]. Для приложений можно выделить два класса смесей: дискретные и непрерывные.

Числовые характеристики вероятностных распределений

Для анализа моделей важную роль играют не только функциональные, но и числовые характеристики распределений. Согласно [19], [73], для случайной величины ξ со значениями из $\mathbb{R}$ можно выделить следующие характеристики.

- *Меры центральной тенденции.*

  Описывают, вокруг какого значения сконцентрированы реализации случайной величины;

  - *Математическое ожидание.* Для случайной величины ξ, её математическое ожидание определяется равенством (mean);

$$\mathbb{E}[\xi] = \int_{\mathbb{R}} x\mathbb{P}_{\xi}(dx) \qquad \text{(mean)}$$

  - *Медиана.* Для случайной величины ξ, её медиана определяется как множество всех значений m, таких что $F_{\xi}(m) = \frac{1}{2}$, иначе говоря, медиана определяется равенством (med).

$$\mathrm{Med}[\xi] = F_{\xi}^{-1}\left(\tfrac{1}{2}\right) \qquad \text{(med)}$$

    В некоторых случаях, в качестве медианы берут какое-то конкретное значение из множества $F_{\xi}^{-1}(\frac{1}{2})$, такое значение называется *точной медианой* [73];
  - *Мода.* Для случайной величины ξ её мода определяется равенством (mode),

$$\mathrm{Mode}[\xi] = \mathrm{argmax}_{\mathbb{R}}\, f_{\xi}(x) \qquad \text{(mode)}$$

    где $f_{\xi}(x)$ это функция вероятности, если ξ это дискретная случайная величина, и плотность, если ξ — непрерывная.

- *Меры разброса*

  Указывают на склонность величины отклоняться от своего «центрального» значения;

  – *Дисперсия* и *среднеквадратичное отклонение* определяются равенствами (std) и (std) соответственно;

$$\mathbb{D}[\xi] = \mathbb{E}\big[(\xi - \mathbb{E}[\xi])^2\big]; \tag{var}$$
$$\mathrm{std}[\xi] = \sqrt{\mathbb{D}[\xi]} \tag{std}$$

  – *Среднее абсолютное отклонение* определяется (mad).

$$\mathrm{mad}[\xi] = \mathbb{E}[|\xi - \mathbb{E}[\xi]|] \tag{mad}$$

  Иногда вместо внутреннего или внешнего математического ожидания берут точную медиану;

  – *Межквартильный размах* определяется равенством (iqr).

$$\mathrm{IQR}[\xi] = \omega_\xi(0.75) - \omega_\xi(0.25) \tag{iqr}$$

- *Меры скоса*

  Меры асимметрии распределения относительно среднего.

  – *Коэффициент скоса* определяется равенством (skew);

$$\mathrm{Skew}[\xi] = \frac{\mathbb{E}\big[(\xi - \mathbb{E}[\xi])^3\big]}{\mathrm{std}^3[\xi]} \tag{skew}$$

  – *Коэффициент скоса Пирсона* определяется pskew;

$$\mathrm{Skew}^{\mathrm{P}}[\xi] = \frac{\mathbb{E}[\xi] - \mathrm{Med}[\xi]}{\mathrm{mad}[\xi]} \tag{pskew}$$

  – *Обобщенный коэффициент скоса Грюневельда* определяется равенством (pskew).

$$\gamma(u) = \frac{\omega_\xi(1-u) + \omega_\xi(u) - 2\omega_\xi\left(\frac{1}{2}\right)}{\omega_\xi(u) - \omega_\xi(1-u)} \quad \tfrac{1}{2} < u < 1 \tag{qskew}$$

- *Меры эксцесса и тяжести хвостов*
  - *Коэффициент эксцесса* измеряет степень «остроты» вершины распределения и определяется равенством (kurt);

$$\operatorname{Kurt}[\xi] = \frac{\mathbb{E}\big[(\xi - \mathbb{E}[\xi])^4\big]}{\operatorname{std}^4[\xi]} - 3 \qquad \text{(kurt)}$$

  - *Квантильный коэффициент эксцесса* является квантильным аналогом стандартного коэффициента эксцесса [59] и определяется равенством (qkurt);

$$\kappa(u, v) = \frac{\omega_\xi(1-u) - \omega_\xi(u)}{\omega_\xi(v) - \omega_\xi(v)}, \quad 0 < u < v < \frac{1}{2} \qquad \text{(qkurt)}$$

  - *Экспонента хвоста* определяется для распределений, функция выживания которых убывает согласно степенному закону, как число α при котором верна асимптотическая эквивалентность (tail-idx).

$$S_\xi(x) \sim x^{-\alpha}, \quad x \to \infty. \qquad \text{(tail-idx)}$$

    В общей ситуации отдельно определяется индекс для левого хвоста и для правого хвоста.

Также, для случайной величины ξ и натурального числа $n \in \mathbb{N}$ определены моменты, центральные моменты (m_n и μ_n соответственно в равенстве (moment)), абсолютные моменты, абсолютные центральные моменты (v_n и ν_n в равенстве (abs-moment)) и факториальные моменты (κ_n в равенстве (fact-moment)) порядка n. На основе этих характеристик можно производить оценку параметров распределения.

$$m_n = \mathbb{E}[\xi^n] \qquad \mu_n = \mathbb{E}[(\xi - \mathbb{E}[\xi])^n] \qquad \text{(moment)}$$
$$v_n = \mathbb{E}\big[|\xi|^k\big], \qquad \nu_n = \mathbb{E}[|\xi - \mathbb{E}[\xi]|^n] \qquad \text{(abs-moment)}$$
$$\kappa_n = \mathbb{E}[\xi(\xi-1)(\xi-2)\dots(\xi-n+1)] \qquad \text{(fact-moment)}$$

Обобщением моментов являются так называемые L-моменты [27] и их квантильные аналоги LQ-моменты [52].

Другое семейство числовых характеристик приходит из области теории информации, см. например монографию [38]. Далее, подразумевается что ξ необязательно вещественнозначная случайная величина, со значениями из некоторого пространства $\mathcal{X}$ и под плотностью подразумевается плотность в смысле (pdf).

- *Энтропия* распределения с плотностью p относительно меры μ определяется равенством (entr);

$$H_r(p) = -\int_{\mathcal{X}} p(x) \log_r p(x)\, d\mu \tag{entr}$$

- *Кросс-энтропия* из распределения с плотностью q в распределение с плотностью p определяется равенством (entr);

$$CE_r(p\|q) = -\int_{\mathcal{X}} p(x) \log_r q(x)\, d\mu \tag{cross-enrt}$$

- *KL-дивергенция* является мерой расхождения между двумя распределениями с плотностями p и q и определятся равенством (kl-div);

$$\mathcal{D}(p\|q) = -\int_{\mathcal{X}} p(x) \log_r \frac{q(x)}{p(x)}\, d\mu \tag{kl-div}$$

Методы из теории информации активно применяются в статистике, см. например [1]. В частности, зачастую рассматривают обобщенный вариант KL-дивергенции — *f-дивергенцию*, которая определяется для любой выпуклой функции $f\colon \mathbb{R}_+ \to \mathbb{R}$ равенством (f-div).

$$\mathcal{D}_f(p\|q) = -\int_{\mathbb{R}} p(x) f\left(\frac{q(x)}{p(x)}\right) dx \tag{f-div}$$

С информационными характеристиками тесно связана *информация Фишера*. Для параметрического семейства плотностей $f(x;\theta), \theta \in \Theta \subseteq \mathbb{R}$, информация Фишера это функция от параметра, задаваемая равенством (FI).

$$\mathcal{I}(\theta) = \mathbb{E}\left[\left(\frac{\partial}{\partial\theta} \log f_\xi(x;\theta)\right)^2\right] \tag{FI}$$

Способы моделирования вероятностных распределений

Для вычисления метрик качества и моделирования поведения вероятностных систем необходимо уметь производить стохастическое моделирование случайных величин. Согласно [26], для генерации выборок можно выделить три основных метода:

- *Метод обратного преобразования*, который базируется на том факте, что для случайной величины ξ с квантильной функцией $\omega_\xi(p)$, распределение случайной величины $\omega_\xi(U), U \sim \mathcal{U}(0;1)$ будет совпадать с распределением ξ;
- *Метод декомпозиции*, который используется для генерации выборок из смешанных распределений. Общая идея заключается в том, что сначала генерируется значение параметра (например номер кластера), а затем уже случайная величина при условии зафиксированного значения параметра;
- *Метод отбора (rejection sampling)*, который используют когда предыдущие два метода не могут быть использованы. Этот метод значительно медленнее предыдущих и при его использовании возникает много нюансов, но для его использования необходим доступ только к плотности распределения;

При этом дискретные распределения требуют отдельного рассмотрения. Существуют также методы основанные на методе отбора, для генерации выборок из распределений заданных с помощью интегральных преобразований. При этом, нередки ситуации, плотность распределения $f_\xi(x)$ известна только с точностью до нормализующей константы или если требуется производить генерацию в сложных или многомерных пространствах. Для таких случаев разработаны методы на основе марковских цепей Монте-Карло (MCMC). Суть метода заключается в том, что на пространстве реализаций надо завести некоторое случайное блуждание, которое в пределе будет давать желанное распределение, детали см. например в книге [45].

Примеры вероятностных моделей в PySATL

В заключение этого раздела отметим, что поддержка работы с представленными ранее объектами является необходимой для PySATL в рамках существующих и будущих проектов. Пакет MPEst[8] использует различные числовые характеристики, такие как L-моменты для оценок параметров в моделях смеси. Пакет NMVMEstimation[9] занимается специальными видами непрерывных смесей и использует различные интегральные преобразования. Библиотека Experiment[10] использует базовые характеристики распределений для оценок мощностей статистических тестов методом Монте-Карло. С помощью арифметики распределений можно будет получать точные распределения статистик используемых при проверке гипотез В ближайшем будущем планируется начать разработку библиотек для регрессионного анализа и оценки параметров, где также широко потребуется использование различных свойств и характеристик распределений.

Обзор литературы

В этом разделе рассматривается литература освещающая различные алгоритмические и программные аспекты при работе с вероятностными распределениями. Большая часть такой литературы фокусируется на дискретных случайных величинах и/или непрерывных одномерных случайных величинах. В многомерном случае для вычисления характеристик и/или симуляции зачастую необходимо считать кратные интегралы и производить довольно сложные с алгоритмической точки зрения действия, что значительно осложняет анализ качества вычислительных алгоритмов, особенно в ситуации когда истинные (с точки зрения математики) значения недоступны. Как отмечено в монографии [66], общее состояние знаний касающихся вычислений характеристик в многомерном случае «остается просто набором методов для узкоспе-

[8] https://github.com/PySATL/MPEst (дата обращения: 9 января 2025 г.).
[9] https://github.com/PySATL/PySATL_NMVM_Module (дата обращения: 9 января 2025 г.).
[10] https://github.com/PySATL/pysatl-experiment (дата обращения: 9 января 2025 г.).

циализированных ситуаций в сочетании с немногочисленными общими подходами». Примерно такое же состояние генерации случайных векторов, см. книгу [31]. С той же проблемой сталкиваются случайные геометрические примитивы — некоторый обзор существующих методов для симуляции распределений приведены в статье [33], однако анализ точности не производился.

Вычисление функциональных и числовых характеристик

Несмотря на то, что по численным методам существуют достаточно большое количество литературы, вычисления именно в контексте теории вероятностей, не имеют большого распространения, так как большее число литературы в которой описаны различные методы, работает в предположении доступность идеально точной арифметики. Исключением является книга [51] в которой в общих чертах описаны особенности численной статистики. Впервые, на то что при работе с распределениями, необходимо учитывать особенности машинной арифметики, было обращено внимание в работе [50], ранее вопрос аппроксимации, но не в контексте машинной арифметики, был рассмотрен в [12], где были сформулированы некоторые оценки на погрешности возникающие при вычислении с использованием аппроксимированных распределений.

При этом ситуация когда неаккуратная реализация приводит к ошибкам в вычислениях довольно частая, см. например работу [35], некоторые из таких ошибок в современном ПО были обнаружены относительно недавно[11]. Проблемам в точности в первую очередь подвержены функция распределения и функция выживания, так как плотности распределений, как правило, доступны в терминах элементарных функций. Существует несколько работ нацеленных на анализ точности статистического ПО, см. работы [48], [35], [4], в работе [47] представлена общая методология анализа точности ПО, в частности при вычислении функций распределения и функций выживания предлагают использовать метрику (lre), которая представляет собой количество правильно

[11] https://github.com/scipy/scipy/issues/18117 (дата обращения: 9 января 2025 г.).

вычисленных значащих цифр относительно «истинного» значения v_{true}.

$$\text{LRE}(v; v_{\text{true}}) = -\log_{10}\left|\frac{v_{\text{true}} - v}{v_{\text{true}}}\right| \tag{lre}$$

В качестве v_{true} автор предлагает использовать значения из библиотеки DCDFLIB [9], отмечая её высокую точность.

Другой проблемной характеристикой является квантильная функция. Для вычисления квантилей необходимо решать уравнение обращения, т.е. найти x такое, что $F(x) = p$. Это может быть сделано с помощью бинарного поиска (в силу монотонности функции распределения) или любого другого алгоритма поиска корней. Однако, для этого нужно иметь быструю и достаточно точную процедуру вычисления $F(x)$, которые во многих случаях, как отмечалось ранее, недоступны. В этом случае существуют альтернативные подходы, обзор которых можно найти в работе [46]. Также, в работе [46] предложены две меры для оценки качества вычисления квантильных функций, определяемых равенствами (qe1) и (qe2), и рассмотрены алгоритмические аспекты вычисления квантилей с точки зрения параллельных вычислений. Другой работой, ориентированной на вычисление квантильных функций является [64], в которой рассмотрен подход на основе решения дифференциального уравнения зависящего от плотности.

$$E_1 = \max_u \left|\frac{\hat{\omega}_\xi(u)}{\omega_\xi(u)} - 1\right| \tag{qe1}$$

$$E_2 = \max_u \left|\frac{F_\xi(\hat{\omega}_\xi(u))}{u} - 1\right| \tag{qe2}$$

Числовые характеристики, рассмотренные ранее, являются либо характеристиками моментного типа, такие как дисперсия (var), либо характеристиками квантильного типа, например (iqr). Многие моментные характеристики часто могут быть представлены в виде (m-int) или как алгебраическое выражение от таковых.

$$\int_{\mathbb{R}} T(x)\, dF_\xi(x) \text{ или в случае непрерывной } \xi \int_{\mathbb{R}} T(x) f_\xi(x)\, dx \tag{m-int}$$

Вычисление интегралов вида (m-int) зачастую трудно в силу того что требуется брать интегралы вдоль вещественной прямой и невозможно контролировать погрешность интегрирования. Однако существует ряд техник позволяющих упростить задачу. Например, в [25], использует прием с заменой, при котором в интеграл (m-int) делается подстановка (q-int) и далее применяется подходящая процедура интегрирования с контролируемой погрешностью

$$\int_{\mathbb{R}} T(x)\, dF_\xi(x) \mapsto \int_0^1 T(\omega_\xi(p))\, dp \qquad \text{(q-int)}$$

В отличие от «классических» функциональных характеристик, интегральные преобразования более подробно изучены в рамках численных методов, так как они возникают не только в теории вероятностей. Об особенностях интегральных преобразований существует достаточно много работ, см. [3, 71, 11, 7, 67].

Арифметика случайных величин

Известно, [63], что обычные численные методы для вычисления сверток участвующих в определении арифметических операций являются неподходящими так как зачастую возникают следующие проблемы.

- Зачастую надо уметь вычислять значение плотности сразу в нескольких точках;
- Неконтролируемая погрешность при многократном вычислении арифметических операций;

В целом, согласно [30] можно выделить несколько подходов к вычислению интегралов возникающих в арифметике случайных величин

- Вычисление только в ситуациях, когда есть аналитический результат в доступном виде. Проблема такого подхода заключается в ограниченности, потому что даже в рамках существования численных результатов, многие из них опираются на теорию специальных функций; Отказ от вычисления таких функций сильно сужает и без того малую область применимости такого подхода.

- Использование теории специальных функций. Фундаментальная работа в этом направлении это монография [63], в которой удалось получить представления или аппроксимации с помощью обобщенных гипергеометрических функций. Однако, этот подход не поддерживает сумму и разность случайных величин и требует существования младших моментов у рассматриваемых распределений;

- Использование систем компьютерной алгебры, см. например [22], которое в конечном счете также упирается в невозможность прямого вычисления интегралов и в итоге приводит к численном интегрированию;

- Обыкновенное численное интегрирование. Проблемы этого способа были освещены выше. Однако стоит отметить, что существуют специальные подходы для плотностей, которые убывают определенным образом [16]. Эти методы потенциально могут давать хорошие результаты. К тому же, в силу детерминированности квадрутрных формул, вычисления таких интегралов могут быть выполняться параллельно;

- Использование метода Монте-Карло. Одним из универсальных подходов является генерация выборок из распределений суммы-/произведения и использование этих выборок для оценки интересующих характеристик распределения. Это универсальный метод, который к тому же позволяет получать некоторое представление о погрешности распределений, однако погрешность результата при таком методе убывает со скоростью $1/\sqrt{n}$ [41], что делает невозможным получение точных результатов.

Альтернативой этим подходам является использование вместо самих плотностей различных аппроксимаций или работать с аппроксимированными интегральными преобразованиями, см. [37], [74], и [30] для наиболее перспективного подхода.

В заключение отметим, что изложены выше подходы являются в некотором смысле универсальными и будут работать также для вычис-

ления преобразований над случайными векторами или геометрическими примитивами, однако хороших методов на основе аппроксимаций для таких случаев пока нет. В свою очередь, для дискретных распределений, таких проблем как правило не возникает так как обычно доступны прямые точные (хоть и весьма трудоемкие) вычисления.

Литература по моделированию вероятностных распределений

По всей видимости, первым системным обзор различных методов по генерации вероятностных распределений была работа [13], однако в ней не учтены особенности машинной арифметики. В монографии [36], был рассмотрен вопрос качестве различных подходов к генерации равномерно распределенных на отрезке $[0;1)$ чисел, и были предложены некоторые критерии оценивания таких генераторов. Позднее эти методы были реализованы и дополнены в библиотеке [39].

Вопросы качества генерации случайных чисел, имеющих распределение, отличное от равномерного подробно рассмотрены в [26], где рассматриваются различные алгоритмы для генерации из произвольных распределений, производится их анализ на предмет скорости и точности. Стоит отметить, что анализ скорости производился относительно «стандартного» алгоритма, и относится к исследованию самих алгоритмов, нежели программных реализаций. Для проверки качества генерации выборок из непрерывных распределений авторы предлагают следующий подход. Для реализаций $x_1, \ldots, x_n, \ldots$ случайной величины ξ с функцией распределения $F_\xi(\cdot)$ необходимо вычислить преобразованные значения $u_1 = F_\xi(x_1), \ldots, u_n = F_\xi(x_n)$. Если генератор является качественным, то последовательность $u_1, \ldots, u_n$ должна быть последовательностью независимых реализаций случайной величины, имеющей равномерное распределение на отрезке $[0;1)$. Далее, предлагается использовать стандартные способы проверки качества генерации чисел с равномерным распределением на $[0;1)$.

В работе [77] можно найти погрешности в терминах математических ожиданий которые возникают при моделировании распределений с тяжелыми хвостами (т.е. начиная с какого-то момента абсолютные

моменты распределения равны бесконечности). В работе [76] того же автора, предлагается использовать метод декомпозиции для моделирования таких распределений и обсуждается улучшения, привносимые таким подходом. Другой подход,на данный момент не апробированный, — частичная генерация случайных величин [54], идея которого заключается в том, что битовое представление случайной величины генерируется «лениво», только при прямой необходимости. Потенциально этот подход может предоставить возможность проводить стохастического моделирование со сколь угодно малой погрешностью, за счет увеличения потребления памяти и времени.

В отличие от генерации непрерывных вещественнозначных случайных величин, для дискретных случайных величин вопрос генерации обстоит значительно проще, см. [36] и [26] для подробностей.

Обзор решений

Выделяются следующие группы инструментов.

- Инструменты для алгебраических операций над вероятностными распределениями. Эти инструменты производят численные/символьные вычисления в разных вероятностных моделях и предоставляют основные характеристики распределений;
- Инструменты для базовых задач частотной статистики. Как правило, эти инструменты позволяют вычислять точечные/доверительные оценки для параметров распределения и моделировать выборки из разных распределений;
- Инструменты байесовского вывода. В большинстве случаев, это инструменты на основе метода марковских цепей Монте-Карло или на основе вариационного вывода. Такие инструменты, как правило, не позволяют использовать себя для решения классических задач частотной статистики, так как подразумевают некоторого априорного распределения на пространстве параметров.

Приведенная выше классификация является условной и нужна лишь для того, чтобы дать некоторое представление об текущем состоянии

статистического ПО. Отдельно обратим внимание на то обстоятельство, что инструмента, который бы качественно мог решать задачи, характерные для каждой из выше приведенных групп, пока не существует. Тем не менее системы компьютерной алгебры, такие как Wolfram Mathematica [75] и MATLAB [29], а также их открытые аналоги GNU Octave [15], SageMath [60] и SciLab [62], с одной стороны предоставляют базовую функциональность для всех трех типов задач, а с другой обладают встроенной поддержкой различных математических методов (например, вычисления интегралов или решения оптимизационных задач), которые нужны для реализации более продвинутой функциональности. Таким образом, ядро PySATL можно было бы реализовать на основе этих систем, однако

- SciLab, SageMath и GNU Octave распространяются под лицензией GPLv3, что фактически не делает их совместимыми с любыми коммерческими проектами;
- Напротив, MATLAB, Wolfram Mathematica и другие коммерческие системы распространяются на платной основе, что сильно сузит круг потенциальных пользователей PySATL.

Так как целевая аудитория PySATL состоит как из обычных пользователей и исследователей, так и из коммерческих компаний, для ядра потенциально подходят только библиотеки на основе разрешительных лицензий свободного ПО. Поэтому возможность реализовывать ядро на основе одной из таких систем не рассматривается.

Это означает, что встаёт вопрос выбора языка программирования, на котором будет написано ядро. Можно выделить несколько языков, которые подходят для решения этой задачи:

- Языки имеющие распространение в академической среде: к ним относятся R, Julia, Python;
- Языки ориентированные на высокопроизводительные вычисления и имеющие распространение в коллективе разработчиков PySATL. Это C/C++, Rust.

Так как количество инструментов для работы с распределениями достаточно большое и всех их покрыть невозможно, инструменты рассмат-

риваемые в этой работе должны удовлетворять некоторым критериям.

Во-первых, с точки зрения производительности и точности, не имеет смысл рассматривать инструменты, которые не предоставляют некоторую «базовую функциональность», а фокусируются на каком-то одном аспекте работы с распределениями, например на обращении характеристических функций. Несмотря на то, что анализ таких инструментов был бы весьма интересен, он выходит за рамки данного обзора и возможно будет произведен в будущем, когда будет стоять вопрос добавления реализации соответствующей функциональности в ядре. Во-вторых, в качестве ядра может выступать инструмент который

- либо имеет базовую функциональность, активно поддерживается сообществом (т.е. последнее обновление было менее года назад) и написан на одном из указанных выше языков;
- либо инструмент, который обладает достаточно широкой функциональностью, покрывающей большую часть описанных ранее характеристик и операций над распределениями.

Поясним что понимается под «базовой функциональностью». Это поддержка одномерных непрерывных и дискретных распределений, для которых можно вычислять функцию распределения, плотность или функцию вероятности и квантильную функцию. Такие требования обоснованы тем, что эта функциональность по умолчанию предоставляется языком R, который на данный момент является самым одним из самых популярных языков для разработки статического ПО[12]. Под «достаточно широкой функциональностью» понимается возможность вычислять моменты распределений, производить генерацию и вычислять характеристические функции. На основании этого, можно сформировать следующие критерии для инструментов, обзор которых нужно произвести.

T1 Инструмент предоставляет достаточно широкую функциональность и активно поддерживается;

T2 Инструмент предоставляет базовую функциональность, активно поддерживается и написан на одном из подходящих языков;

[12]В Journal of Statistical Software более 80% публикаций посвящены пакетам на R

Также, имеет смысл рассмотреть инструменты удовлетворяющие следующему критерию.

Т3 Инструмент является узкоспециализированным и не имеет некоторых элементов базовой функциональности.

Инструменты, которые были отнесены к последней категории, в первую очередь предоставляют интерес с точки зрения (архитектурного) моделирования предметной области. В таблице 1 приведен список инструментов, которые подходят под эти критерии.

Инструмент	**Язык**	**Т**
Библиотека SciPy Модуль stats	Python	1
Библиотека NumPy Модуль radnom	Python	3
Библиотека PyTorch Модуль distributions	Python	2
Библиотека PaCAL	Python	3
Библиотека Pomegranate	Python	3
Библиотека PyMC	Python	3
Библиотека Distributions.jl	Julia	2
Проект (система пакетов) Probability Distributions	R	1
Библиотека Boost Модуль distributions	C++	2
Библиотека TensorFlow Модуль distributions	C++	2
Библиотека statrs	Rust	2
Библиотека UNU.RAN	C	3

Таблица 1: Инструменты, отобранные подходящие под критерии Т1-Т3

Подробное сравнение функциональности рассматриваемых инструментов представлено в главе 1 настоящей работы. Ниже представле-

но краткое описание рассматриваемых решений[13]: выделены основные особенности, потенциально сильные и слабые стороны, а также статус проекта (заброшен/активно разрабатывается/другое).

Решения на языке Python

NumPy.random Модуль библиотеки NumPy, используемый для генерации выборок из распределений. Многие другие библиотеки (например, PyMC) используют его, когда предоставляют функциональность генерирования выборок.

- **Особенности**. Реализован в объектно ориентированном стиле, обладает гибкостью в плане добавления алгоритмов генерации и контроля над генерацией случайных битов для этих алгоритмов;
- **Лицензия**. Собственная;
- **Статус**. Проект находится в стабильном состоянии и активно поддерживается;
- **Сильные стороны**:Поддержка широкого класса непрерывных и дискретных распределений, поддержка многопоточной генерации;
- **Слабые стороны**: Нет другой функциональности кроме генерации выборок.

SciPy.stats Библиотека базируется на NumPy для работы с массивами и предоставляет широкие интерфейсы для распределений

- **Особенности**: Реализован в объектно ориентированном стиле, имеет поддержку более 80 распределений и широкую функциональность для них;
- **Лицензия**: BSD 2 Clause;
- **Статус**: Проект активно развивается. Актуальная версия от 3 января 2025 года;
- **Сильные стороны**: Поддержка более 80 распределений, широкая функциональность, простая структура;
- **Слабые стороны**: Слабое покрытие тестами, ограниченные возможности по добавлению новых распределений.

[13]Для первичной структуризации текста ниже и до конца раздела использовался ChatGPT

PyTorch.distributions Библиотека нацеленная на работу с многомерными распределениями для задач машинного обучения. Оптимизирована для автоматического дифференцирования вычислений с полной интеграцией в PyTorch.

- **Особенности**: Интеграция с автоматическим дифференцированием в PyTorch для тензорных вычислений.
- **Статус**: Модуль находится в стабильной стадии, активно дорабатывается документация. Последнее обновление от 8 января 2025 года.
- **Лицензия**: Собственная;
- **Сильные стороны**: Поддержка преобразований распределений, широкого класса числовых характеристик, использование GPU и автоматического дифференцирования для вычислений.
- **Слабые стороны**: Узкая направленность на задачи глубокого обучения.

Pomegranate Ориентирована на вероятностное моделирование и композицию статистических моделей, с использованием автоматического дифференцирования.

- **Особенности**: Гибкость при создании вероятностных моделей, и использование GPU для их оценивания.
- **Лицензия**: MIT;
- **Статус**: Инструмент находится в стабильной стадии развития. В июле 2024 состоялся выпуск версии 1.1.
- **Сильные стороны**: Гибкость, простота при работе с существующими моделями.
- **Слабые стороны**: Высокий порог входа для расширения, малая функциональность в контексте стандартных распределений.

PaCAL Предоставляет инструменты для вычислений преобразований над распределениями.

- **Особенности**: Широкие возможности для арифметики и преобразованиям распределений. Исследована точность и корректность реализации.
- **Статус**: Заброшен.

- **Лицензия**: GPLv3;
- **Сильные стороны**: Поддержка всех основных операций для работы с одномерными непрерывными распределениями.
- **Слабые стороны**: Отсутствие качественной документации, не проведено тестирование.

PyMC Полностью заточенная под байесовский вывод библиотека. Использует собственный движок для автоматического дифференцирования и тензорных вычислений.

- **Особенности**: Ориентированность на декларативный подход в определении математических моделей и использование Марковских цепей Монте-Карло для их анализа.
- **Статус**: Активно развивающийся проект с большой пользовательской базой. Последнее версия от 8 января 2025 года.
- **Лицензия**: Apache License 2.0;
- **Сильные стороны**: Интерфейсы направленные на легкое создание собственных моделей для данных.
- **Слабые стороны**: Узкая специализация исключительно под байесовские вычисления. Большие накладные расходы при выполнении некоторых операций над распределениями.

Решения на Julia и R

Distributions.jl Библиотека на языке Julia для работы с вероятностными распределениями, использующая основные возможности языка.

- **Особенности**: глубокая система типов и использование мультиметодов позволяет охватить широкий класс задач при работе с вероятностными распределениями;
- **Статус**: Проект находится в стабильном состоянии. Актуальная версия от 8 января 2025 года;
- **Лицензия**: MIT;
- **Сильные стороны**: Широкая функциональность и высокая скорость работы кода;
- **Слабые стороны**: Отсутствие качественного тестирования.

R Distributions Набор пакетов для языка R предоставляющий функциональность для работы с распределениями

- **Особенности**: Практически полностью выполнен в процедурном стиле;
- **Статус**: Активно обновляется, большинство пакетов были имеют соответствующие публикации;
- **Лицензия**: Так как проект состоит из разных пакетов, присутствуют разные лицензии, в том числе и из семейства GPL;
- **Сильные стороны**: Широкая функциональность для базовых распределений;
- **Слабые стороны**: Процедурный стиль осложняет создание больших проектов. Низкая общность кода. Отсутствие единообразности в лицензировании.

Решения на языках C, C++ и Rust

UNU.RAN Библиотека посвященная различным алгоритмам генерации выборок.

- **Особенности**: Фокусируется на различных методах автоматической генерации выборок;
- **Статус**: Проект находится в зрелой фазе. Последнее обновление было два года назад;
- **Лицензия**: GPLv2;
- **Сильные стороны**: Широта доступных алгоритмов и наличие сопутствующего анализа;
- **Слабые стороны**: Узкая направленность.

Boost Distributions . Часть библиотеки Boost, нацеленная на предоставление базовых характеристик основных распределений и генерации выборок из них.

- **Особенности**: Шаблонная структура, предоставляющая высокий контроль над базовой функциональностью;
- **Статус**: Активно поддерживаемый проект;
- **Лицензия**: собственная;

- **Сильные стороны**: Гибкость, поддержка пользовательских распределений и контроль над точностью вычислений;
- **Слабые стороны**: Высокий порог входа.

TensorFlow Distributions Часть вычислительного графа TensorFlow, оптимизирована для дифференцируемого программирования.

- **Архитектура**: Интеграция с TensorFlow для градиентного обучения;
- **Статус**: Модуль находится в стабильной стадии, активно дорабатывается документация; Актуальная версия от 8 ноября 2024 года;
- **Лицензия**: Apache License 2.0;
- **Сильные стороны**: Поддержка широкого класса одномерных и многомерных распределений, вычисления их числовых характеристик, использование GPU и автоматического дифференцирования для вычислений;
- **Слабые стороны**: Высокие накладные расходы для обычных задач, специализация на задачах машинного обучения.

statrs Библиотека для статистических вычислений и поддержки распределений на чистом Rust.

- **Особенности**: Выполнен в объектно-ориентированном стиле и предоставляет базовую функциональность;
- **Статус**: Проект еще не является стабильным, и активно развивается. Актуальная версия от 8 июля 2024 года;
- **Лицензия**: MIT;
- **Сильные стороны**: Сильный движок для генерации выборок;
- **Слабые стороны**: Меньшая экосистема по сравнению с Python.

Инструменты, не вошедшие в список

Необходимо сказать об инструментах, которые не подошли ни по одному из критериев, однако предоставляют самостоятельный интерес. Одним из таких инструментов является написанная на C библиотека GNU Scientific Library (GSL). В ней доступная базовая функцио-

нальность для распределений и генерация случайных величин, в том числе из урезанных и цензурованных. Однако команда проекта сейчас занята другими частями библиотеки и модуль с распределениями не получал обновления уже более двух лет. Среди библиотек реализованных на Java, интерес представляет библиотека Apache Commons Statistics. Однако её функциональность остаётся в рамках стандартных статистических задач, и она не предоставляет обширных возможностей для сложного моделирования. Ещё одним примером из экосистемы Java является Colt — библиотека для научных вычислений, однако поддержка статистики не покрывает даже базовой функциональности.

В среде .NET существует несколько библиотек предлагающих некоторые методы для работы с вероятностными распределениями, однако в лучшем случае они покрывают лишь базовую функциональность, необходимую для ядра. Среди них можно выделить вдохновлённую NumPy библиотеку SciSharp, Math.NET и Accord.NET, FSharp.Stats.

Аналогично дело обстоит с библиотеками на Haskell, однако стоит отметить библиотеку Statistics, как интересный пример реализации статистического ПО в функциональном стиле. Библиотеки для Fortran, используемые в научных вычислениях, также предоставляют разнообразные инструменты для работы с распределениями, все в рамках базовой функциональности. Исключение составляется библиотека afnl, которая предоставляет достаточно широкую функциональность, но распространяется под лицензией GPLv-2 и более 8 лет не получала обновлений.

Выводы

При работе с вероятностными распределениями можно выделить следующие крупные группы задач:

1. Вычисление числовых и функциональных характеристик вероятностных распределений.

2. Генерация выборок с заданным распределением.

3. Численные методы для арифметики и преобразования вероятностных распределений.

4. Манипуляции с семействами вероятностных распределений.

Существует множество инструментов реализующих базовую функциональность и какую-то функциональность из той или иной группы задач, однако универсального инструмента, который бы мог решать задачи из всех групп выше, не существует. При этом, для вычислительного ядра в рамках проекта PySATL необходимо иметь возможность производить действия с распределениями, описанные в рамках обзора предметной области. Это означает что нужно провести сравнение инструментов, которые могут потенциально быть реализованы в качестве ядра.

К таким инструментам можно отнести инструменты подходящие под критерий T1 или T2 (то есть имеющие минимальную функциональность): это библиотеки SciPy, R, Distributions.jl, Boost, TensorFlow, PyTorch и statrs. Необходимо произвести сравнение функциональности этих библиотек, чтобы заключить, можно ли на основе какой-то из них разработать ядро для проекта PySATL.

1 Сравнение инструментов

Этот раздел поделен на три части

- В разделе «сравнение функциональности» представлен ряд критериев для оценки функциональности инструментов и произведено сравнение инструментов по этим критериям;
- В разделе «сравнение особенностей реализации» представлены ряд критериев, оценивающих расширяемость инструмента и произведено сравнение инструментов по этим критериям;
- В разделе «выводы» представлен взгляд на результаты сравнения с архитектурной точки зрения.

1.1 Сравнение функциональности

В таблице 2 отображена информация, о том, какие типы распределений какой инструмент поддерживает. Выделены следующие степени поддержки.

- Широкая — соответствующий тип является самостоятельной сущностью в рамках инструмента.
- Частичная — представлено хотя бы одно семейство соответствующего типа.

В библиотеках PyTorch и TensorFlow все четыре типа распределений неразличимы между собой, но так как преимущественно представлены одномерные непрерывные и дискретные распределения, для этих типов указана широкая поддержка.

	Дискретные	Одномерные непрерывные	Многомерные непрерывные	Геометрические примитивы
SciPy	Широкая	Широкая	Широкая	Частичная
R	Широкая	Широкая	Широкая	Широкая
Distributions.jl	Широкая	Широкая	Частичная	Частичная
Boost	Широкая	Широкая	Нет	Нет
TensorFlow	Широкая	Широкая	Частичная	Нет
PyTorch	Широкая	Широкая	Частичная	Нет
statrs	Широкая	Широкая	Частичная	Нет

Таблица 2: Степень поддержки различных видов распределений

В таблице 3 отображена информация о способах образования новых распределений из уже существующих. В экосистеме R, в силу того что большинство пакетов реализовано в процедурном стиле, определены частные случаи рассмотренных выше способов, но общей функциональности практически нет, поэтому в таблице указана частичная поддержка операций. В пакете Distributions.jl представлена поддержка арифметики распределений только в случае, когда есть явный аналитический результат операции.

	Смеси	Урезание и цензурирование	Арифметика	Преобразования	Порядковые статистики
SciPy	Дискретные	Есть	Нет	Нет	Есть
R	Частично	Частично	Частично	Частично	Частично
Distributions.jl	Дискретные	Есть	Частично	Нет	Есть
Boost	Нет	Нет	Нет	Нет	Нет
TensorFlow	Дискретные	Нет	Нет	Есть	Нет
PyTorch	Дискретные	Нет	Нет	Есть	Нет
statrs	Нет	Нет	Нет	Нет	Нет

Таблица 3: Степень поддержки различных действий над распределениями

В таблице 4 представлен общий обзор функциональности инструментов, степень широты функциональности определялась в соответствии с критериями представленными в таблицах 5 для работы с функциональными характеристиками, 6 для работы с числовыми характеристиками и 7 для работы с генерацией выборок.

	Функциональные характеристики (КФ)	**Числовые характеристики (КЧ)**	**Генерация выборок (КГ)**
SciPy	4	6	3
R	3	4	2
Distributions.jl	4	6	1
Boost	2	3	1
TensorFlow	2	4	1
PyTorch	2	4	1
statrs	2	2	1
max	**4**	**6**	**3**

Таблица 4: Сравнение инструментов по функциональности

Код	**Описание критерия**	**Комментарий**
1	Доступны pdf/pmf, cdf, ppf	
2	Выполнен КФ-1 и доступна еще одна функциональная характеристика	Возможно, не для всех распределений
3	Выполнен КФ-2 и есть доступ к интегральным преобразованиям	Возможно, не для всех распределений
4	Выполнен КФ-3 и есть возможность автоматического вывода характеристик	

Таблица 5: Уровни поддержки функциональных характеристик

Код	**Описание критерия**	**Комментарий**
1	Доступны математическое ожидание и дисперсия	
2	Выполнен КФ-1 и доступна еще хотя бы одна числовая характеристика	Возможно, не для всех распределений
3	Выполнен КФ-2 и доступна еще энтропия	Возможно, не для всех распределений
4	Выполнен КФ-3 и есть вычисление KL дивергенции	Возможно, не для всех распределений
5	Выполнен КФ-2 и есть автоматический вывод характеристик	
6	Выполнен КФ-4 и есть автоматический вывод характеристик	

Таблица 6: Уровни поддержки числовых характеристик

Код	**Описание критерия**	**Комментарий**
1	Доступна генерация выборок из заранее преоопредленного набора распределений	
2	Выполнен КФ-1 и доступна генерация из пользовательских распределений	
3	Выполнен КФ-1 и доступна настройка параметров алгоритма генерации	

Таблица 7: Уровни поддержки генерации выборок

1.2 Сравнение особенностей реализации

Помимо сравнения существующей функциональности необходимо рассмотреть особенности предоставления доступа к ней и возможность добавления новой функциональности.

Можно выделить три подхода к рассматриваемым библиотекам

- Объектно-ориентированный, в котором распределения являются объектами, как правило, неизменяемыми;
- Процедурный, в котором распределения как сущности не представлены ни в каком виде, и все оперирование происходит с функциональными характеристиками, определёнными с помощью уникальных функций для каждого вида распределений;
- На основе множественной диспетчеризации, в котором распределения являются структурами без методов, а вычисление характеристик и генерации распределений делается на основе мультиметодов. В этом случае разработчик просто определяет необходимые перегрузки процедур вычисления.

Библиотеки SciPy, Boost, TensorFlow, PyTorch и statrs используют объектно-ориентированный подход, с некоторыми небольшими различиями. Общие принципы такого подхода следующие

- Семейство распределений — это класс, наследующийся от некоторого базового для всех распределений класса, его объекты это конкретные распределения;
- Доступ к функциональным и числовым характеристикам распределений осуществляется через методы класса.

При таком подходе добавление нового семейства распределений производится с помощью наследования. В SciPy для этого достаточно определить либо CDF, либо PDF, и остальная функциональность станет доступна автоматически. Листинг 1 показывает как это сделать. После

исполнения этого кода, становятся доступны все методы доступные в SciPy, в том числе и автоматическая генерация выборок.

В TensorFlow, PyTorch и Boost, также необходимо наследоваться, однако, в отличие от SciPy необходимо полностью реализовывать все методы класса. Пример определения класса распределения в библиотеке Boost доступен в [18]

Листинг 1.: Добавление в библиотеку SciPy нового распределения

```
from scipy.stats import rv_continuous
class SqrtDistr(rv_continuous):
    def _cdf(self, x, max):
        self.max=max
        return 0 if x < 0 else min(1, math.sqrt(x/ max))
sqrt_distr=SqrtDistr(name="sqrt of uniform")
```

В библиотеке Distributions.jlиспользуется подход на основе мультиметодов. В силу особенности языка Julia, архитектура библиотеки Distributions.jl представляет собой дерево абстрактных типов, в листьях которого находятся конкретные типы, являющиеся (по умолчанию) неизменяемыми структурами. Несмотря на то, что добавление нового распределения требует реализации вычисления его функциональных характеристик, можно указать способ вычисления по умолчанию с помощью расширения метода, как это сделано в листинге 2. После исполнения этого кода, при создании одномерного равномерного распределения, если для него не определена функция плотности, будет использоваться численное дифференцирование определенное в листинге 2. При этом если определен другой способ вычисления плотности, численное дифференцирование использоваться не будет. Аналогично работает добавление числовых характеристик.

Листинг 2.: Добавление в библиотеку Distributions.jl вычисления плотности через численное дифференцирование

```
using Random, Distributions
import Distributions.pdf
function pdf(d::ContinuousUnivariateDistribution, x::Real)
    return (cdf(d, x + 0.1) - cdf(d, x - 0.1)) / 0.2
end
```

Информацию Сложность расширения для каждого решения представлена в таблице. Поясним что понимать под разными степенями сложностями. Для добавления новых числовых характеристик и новых алгоритмов генерации степень

- Расширение имеет простую степень сложности если для его реализации не нужно модифицировать код библиотеки и не требуется производить действий, не относящихся напрямую к реализации функциональности;
- Расширение имеет среднюю степень сложности если выполнено ровно одно из следующих условий
  - для его реализации необходимо модифицировать код библиотеки без дополнительных накладных действий;
  - для его реализации не нужно модифицировать код библиотеки, но помимо самой реализации функциональности нужно выполнить дополнительные действия напрямую относящиеся к данной функциональности. При этом сложность накладных действий не зависит от числа реализованных характеристик/алгоритмов генерации/распределений;
- Во всех остальных случаях — высокая степень сложности.

Так например, при добавлении распределения в библиотеку Boost необходимо реализовать машинерию связанную с обобщенным програм-

мированием, причем размер этой машинерии зависит от того, сколько у распределения числовых характеристик [18], поэтому в таблице отмечена высокая степень сложности для добавления распределения.

	Сложность добавления нового распределения	**Сложность добавления новой характеристики**	**Сложность добавления нового алгоритма генерации**
SciPy	Простая	Средняя	Простая
R	Средняя	Высокая	Средняя
Distributions.jl	Простая	Простая	Простая
Boost	Высокая	Высокая	Высокая
TensorFlow	Средняя	Средняя	Высокая
PyTorch	Средняя	Средняя	Высокая
statrs	Средняя	Высокая	Высокая

Таблица 8: Сравнение инструментов по расширяемости

Отдельно отметим что при работе с семействами распределений иногда полезно считать, что какой-то параметр имеет конкретное значение, что в таком подходе возможно только с помощью наследования — для разового использования такой подход имеет слишком много накладных расходов, особенно если вокруг класса есть много инфраструктуры, как например в Boost. В SciPy такая возможность присутствовала ранее, но сейчас от нее отказались [28].

В SciPy неявно понятия семейства локации и масштаба и экспоненциального семейства, так как у всех параметрических семейств должны присутствовать параметры локации и масштаба. Вторым спорным решением является то что на пользовательском уровне идет работа только с объектами — за создание объектов распределений отвечает специальный объект того же класса (например в листинге 1 `sqrt_disrt` это объект класса `SqrtDisrt`, но доступ к функциональности осуществляется через этот объект).

1.3 Выводы

Для ядра PySATL не подходит ни один рассмотренных из инструментов и требуется разработка своей библиотеки, возможно на базе одной из существующих.

Несмотря на то, что R обладает большой функциональностью, расширение возможностей экосистемы пакетов R слишком тяжело из-за

процедурного стиля и сведение функциональности этой системы в одно место слишком тяжело. Дополнительным фактором в пользу отказа от R, является отмеченная в обзоре сложная ситуация с лицензиями.

Объектно-ориентированный подход реализованный в библиотеках SciPy, Boost, TensorFlow, PyTorch и statrs накладывает ограничения на расширяемость. Из всех инструментов реализованных в рамках такого подхода наиболее подходящим является SciPy, особенно с учетом того что он уже активно используется в рамках проекта PySATL. Однако, как отмечено ранее, разработчиками SciPy принято несколько неудачных решений, которые могут привести к проблемам.

При этом, как показывает библиотека Distributions.jl, подход на основе мультиметодов выглядит перспективным, однако ввиду того что на основе этого подхода представлена только одна библиотека, неясно является ли использование подхода на основе мультиметодов приемлемым для разработки ядра. Чтобы принять окончательное решение, необходимо разработать прототип ядра с использованием этого подхода. Ввиду того что в библиотеке Distributions.jl не выявлено каких-либо решений, противоречащих требованиям к ядру, предлагается разработать прототип ядра, расширив библиотеку Distributions.jl

2 Требования к ядру PySATL

2.1 Диаграмма вариантов использования

На основе обзора составлена диаграмма сценариев использования ядра, изображенная на рис. 2

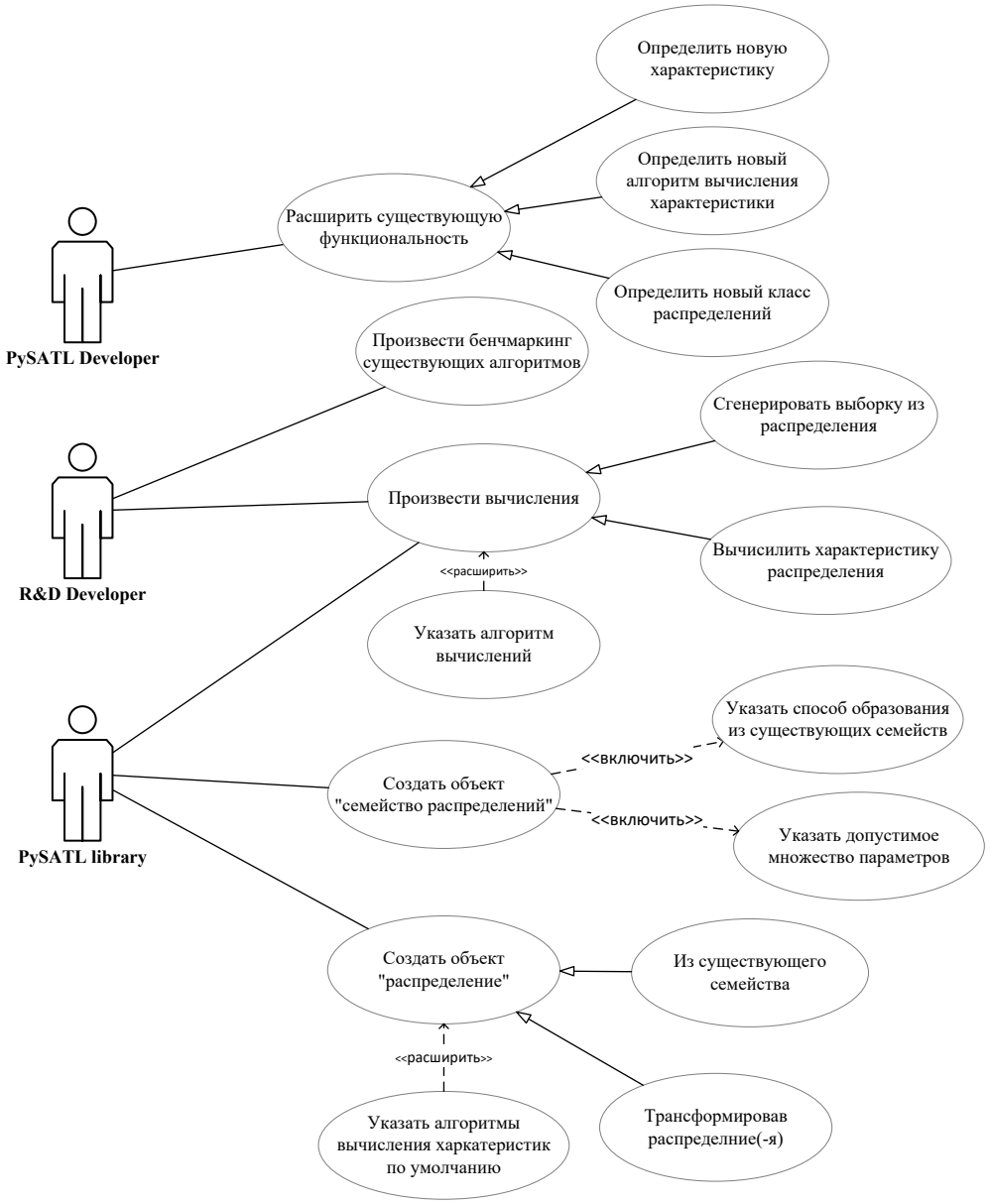

Рис. 2: Диаграмма вариантов использования ядра PySATL

Акторы представленные на диаграмме рис. 2 преследуют разные цели при использовании ядра:

- Актор PySATL Developer заинтересован в том, чтобы использовать ядро для работы с распределениями, которые приходят из его предметной области. Для него является важной возможность создавать свои семейства распределений или новые характеристики для них.

- Актор R&D Developer использует ядро для своих исследовательских нужд. Это может быть поиск подходящего алгоритма с помощью бенчмаркинга или какие-то разовые вычисления

- Актор PySATL Library представляет собой инструмент в рамках PySATL который работает с ядром и уже существующей в нем функциональностью (возможно созданной актором PySATL Developer). Для этого актора важно получать сущности которые он потом может в дальнейшем использовать по своему усмотрению и производит какие-то вычисления.

2.2 Функциональные требования

В рамках этого раздела под корректным вычислением подразумевается выдача типа `NaN` в случае когда вычисляемая величина не определена, и генерация предупреждения. Также требуется, чтобы всякий алгоритм вычисления, в ситуации когда существование результата неоднозначно, прямо сообщал об этом, выдавая предупреждение. При этом подразумевается, что пользователь всегда может потребовать чтобы вместо предупреждения генерировалась ошибка.

Если какой-то алгоритм имеет параметры, у пользователя должна быть возможность эти параметры настроить. К параметрам алгоритмов относятся метод аппроксимации распределений, метод генерации псевдослучайных чисел на отрезке $[0; 1)$ и т.д. При наличии нескольких алгоритмов для получения одного и того же результата, должен быть

определен алгоритм вычисления который используется по умолчанию; при этом пользователь должен иметь возможность

- указать, какой алгоритм использовать;
- добавлять свои собственные алгоритмы вычисления

Ответственность за корректное поведение пользовательских алгоритмов должна лежать на пользователе.

2.2.1 Требования к числовым характеристикам

Для любого распределения должно быть доступно корректное вычисление всех характеристик моментного, квантильного и информационного типа; Пользователь должен иметь возможность добавлять вычисление собственных числовых характеристик.

2.2.2 Требования к функциональным характеристикам

Для любого дискретного распределения должно быть доступно вычисление функции вероятности, для любого непрерывного вещественнозначного распределения — функции плотности. Для распределений геометрических примитивов должна быть возможность вычислять функцию плотности, если она определена;

Для любого вещественнозначного или векторозначных распределений должно быть доступно вычисление функции распределения и функции выживания. Для всех одномерных распределений должно быть доступно вычисление квантильной функции. Также должна быть возможность корректно вычислять характеристическую функцию, производящую функцию моментов преобразование Лапласа, преобразование Меллина и обратные преобразования.

Пользователь должен иметь возможность добавлять вычисление собственных функциональных характеристик

2.2.3 Требования к операциям над распределениями

Для одномерных непрерывных распределений должно быть доступно вычисление распределения получающегося в результате применения

гладкой биекции и иметь поддержку всех операций, представленных в библиотеке PaCAL. Для вещественнозначных дискретных распределений с конечным множеством значений должна быть доступна такая же функциональность.

Для всех операций необходимо иметь возможность добавить в систему аналитический результат, чтобы при вычислении использовался именно он, а не приближенные методы.

Должна быть возможность образовывать урезанные и цензурированые распределения из любых вещественнозначных распределений. Пользователь должен иметь добавлять свои собственные операции над распределениями.

2.2.4 Требования к семействам распределений

Для любого семейства распределений должна быть возможность создать новое семейство распределений, получаемое из исходного частичной подстановкой значений параметров; Над семействами необходимо уметь производить операции образования смесей: непрерывных и дискретных;

Для любого семейства распределений должна быть определена каноническая параметризация, и должна быть возможность добавления новой параметризации. При добавлении новой параметризации, пользователь должен указать как параметры добавляемой параметризации выражаются через каноническую параметризацию и наоборот

2.2.5 Требования к генерации выборок

Для любого распределения, за исключением относящихся к распределениям геометрических примитивов, должна быть возможность генерации выборок. При создании пользовательского распределения должна предоставлять методы для генерации выборок без необходимости реализовывать алгоритмы генерации.

2.2.6 Требования к системе бенчмаркинга

Должен быть доступен следующий сценария системы.

- Пользователь выбирает наборы распределений и наборы параметров, на которых выполняется бенчмарк.
- Пользователь выбирает процедуры, бенчмаркинг которых будет происходить. Пользователь должен уметь использовать процедуры из других пакетов. В частности из пакетов NumPy и UNU.RAN для генерации выборок и SciPy для вычисления числовых характеристик.
- Пользователь выбирает процедуру вычисления метрики качества.
- Пользователь выбирает замерять ли потребление времени и память для используемых алгоритмов.
- Пользователь настраивает формат отчета, запускает и получает результаты бенчмарка.

В качестве метрики точности должны быть реализованы lre (lre). Для квантильных функций также дополнительно должны быть реализованы (qe1) и (qe2).

Заключение

В ходе учебной практики были выполнены следующие задачи

1. Выработаны критерии для оценивания функциональности и расширяемости инструментов работы с вероятностными распределениями. Для семи отобранных инструментов произведено их сравнение по выработанным критериям.
2. Проведен обзор литературы, освещающей основные вопросы измерения качества и корректности ПО для работы с вероятностными распределениями. Предложена методология бенчмаркинга для семплирования и вычисления характеристик.

3. Сформулированы функциональные требования к ядру PySATL.

Разработка бенчмарк-системы была вынесена в следующий семестр. Было принято решение, что бенчмаркинг является частью функциональности ядра, в силу нетривиальной методологии. В результате сравнения инструментов, принято решение о разработке прототипа ядра на языке Julia с использованием библиотеки Distributions.jl в качестве основы.

Список литературы

[1] Shun-ichi Amari. *Information geometry and its applications*. Т. 194. Springer, 2016.

[2] Irina A Antipova. «Inversion of multidimensional Mellin transforms». В: *Russian Mathematical Surveys* 62.5 (2007), с. 977.

[3] David H Bailey и Paul N Swarztrauber. «A fast method for the numerical evaluation of continuous Fourier and Laplace transforms». В: *SIAM Journal on Scientific Computing* 15.5 (1994), с. 1105—1110.

[4] Sai Santosh Bangalore, Jelai Wang и David B Allison. «How accurate are the extremely small P-values used in genomic research: an evaluation of numerical libraries». В: *Computational statistics & data analysis* 53.7 (2009), с. 2446—2452.

[5] Ole Barndorff-Nielsen. *Information and exponential families: in statistical theory*. John Wiley & Sons, 2014.

[6] Ole E Barndorff-Nielsen, Fred Espen Benth, Almut ED Veraart и др. *Ambit stochastics*. Т. 88. Springer, 2018.

[7] Harald Bohman. «Numerical inversions of characteristic functions». В: *Scandinavian Actuarial Journal* 1975.2 (1975), с. 121—124.

[8] Richard Breen. *Regression models: Censored, sample selected, or truncated data*. 111. Sage, 1996.

[9] B Brown, James Lovato и K Russell. «Library of Fortran Routines for Cumulative Distribution Functions, Inverses, and Other Parameters». В: *University of Texas* (1997).

[10] Satish Chandra. «On the Mixtures of Probability Distributions». В: *Scandinavian Journal of Statistics* 4.3 (1977), с. 105—112.

[11] Mark Craddock, David Heath и Eckhard Platen. *Numerical inversion of Laplace transforms: a survey of techniques with applications to derivative pricing*. Тех. отч. University of Technology Sydney, 1999.

[12] Luc Devroye. «A note on approximations in random variate generation». В: *Journal of Statistical Computation and Simulation* 14.2 (1982), с. 149—158.

[13] Luc Devroye. «Nonuniform random variate generation». В: *Handbooks in operations research and management science* 13 (2006), с. 83—121.

[14] David L Donoho и др. «High-dimensional data analysis: The curses and blessings of dimensionality». В: *AMS math challenges lecture* 1.2000 (2000), с. 32.

[15] John W. Eaton и др. *GNU Octave version 5.2.0 manual: a high-level interactive language for numerical computations*. 2020. URL: https://www.gnu.org/software/octave/doc/v5.2.0/.

[16] Sven Ehrich. «On stratified extensions of Gauss–Laguerre and Gauss–Hermite quadrature formulas». В: *Journal of computational and applied mathematics* 140.1-2 (2002), с. 291—299.

[17] William Palin Elderton и Norman Lloyd Johnson. «Systems of frequency curves». В: *(No Title)* (1969).

[18] *Example of creating distribution in Boost*. https://www.boost.org/doc/libs/1_84_0/boost/math/distributions/gamma.hpp.

[19] Felix Famoye. *Continuous univariate distributions, volume 1*. 1995.

[20] Alessio Figalli. «On the continuity of center-outward distribution and quantile functions». В: *Nonlinear Analysis* 177 (2018), с. 413—421.

[21] Janos Galambos и Italo Simonelli. *Products of random variables: applications to problems of physics and to arithmetical functions*. CRC press, 2004.

[22] Andrew G Glen, Diane L Evans и Lawrence M Leemis. «APPL: A probability programming language». В: *The American Statistician* 55.2 (2001), с. 156—166.

[23] Marc Hallin и Dimitri Konen. «Multivariate Quantiles: Geometric and Measure-Transportation-Based Contours». В: *Applications of Optimal Transport to Economics and Related Topics*. Springer, 2024, с. 61—78.

[24] Daniel Hernandez-Stumpfhauser, F. Jay Breidt и Mark J. van der Woerd. «The General Projected Normal Distribution of Arbitrary Dimension: Modeling and Bayesian Inference». В: *Bayesian Analysis* 12.1 (2017), с. 113—133.

[25] Erik Hintz, Marius Hofert и Christiane Lemieux. «Multivariate normal variance mixtures in R: the R package nvmix». В: *Journal of Statistical Software* 102 (2022), с. 1—31.

[26] Wolfgang Hörmann, Josef Leydold и Gerhard Derflinger. *Automatic nonuniform random variate generation*. Springer Science & Business Media, 2013.

[27] JRM Hosking. «L-moments». В: *Wiley StatsRef: Statistics Reference Online* (2014), с. 1—8.

[28] *How to create a new continuous distribution. Before Implementation.* https://scipy.github.io/devdocs/dev/contributor/adding_new.html#adding-a-new-statistics-distribution.

[29] The MathWorks Inc. *Statistics and machine learning toolbox*. Natick, Massachusetts, United States, 2022. URL: https://www.mathworks.com/help/stats/index.html.

[30] Szymon Jaroszewicz и Marcin Korzeń. «Arithmetic operations on independent random variables: a numerical approach». В: *SIAM Journal on Scientific Computing* 34.3 (2012), A1241—A1265.

[31] Mark E Johnson. *Multivariate statistical simulation: A guide to selecting and generating continuous multivariate distributions*. Т. 192. John Wiley & Sons, 1987.

[32] Thomas W Keelin. «The metalog distributions». В: *Decision Analysis* 13.4 (2016), с. 243—277.

[33] John T Kent, Asaad M Ganeiber и Kanti V Mardia. «A new unified approach for the simulation of a wide class of directional distributions». В: *Journal of Computational and Graphical Statistics* 27.2 (2018), с. 291—301.

[34] David G Kleinbaum и Mitchel Klein. *Survival analysis a self-learning text*. Springer, 1996.

[35] Leo Knüsel. «On the accuracy of statistical distributions in Microsoft Excel 97». В: *Computational Statistics & Data Analysis* 26.3 (1998), с. 375—377. ISSN: 0167-9473. DOI: https://doi.org/10.1016/S0167-9473(97)81756-2. URL: https://www.sciencedirect.com/science/article/pii/S0167947397817562.

[36] Donald E Knuth. *The Art of Computer Programming: Seminumerical Algorithms, Volume 2*. Addison-Wesley Professional, 2014.

[37] John E Kolassa. *Series approximation methods in statistics*. Т. 88. Springer Science & Business Media, 2006.

[38] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[39] Pierre L'ecuyer и Richard Simard. «TestU01: AC library for empirical testing of random number generators». В: *ACM Transactions on Mathematical Software (TOMS)* 33.4 (2007), с. 1—40.

[40] Arnaud de La Fortelle. «A study on generalized inverses and increasing functions Part I: generalized inverses». working paper or preprint. Авг. 2015. URL: https://minesparis-psl.hal.science/hal-01255512.

[41] Nicolas Lanchier. *Stochastic modeling*. Springer, 2017.

[42] Lawrence M Leemis и др. «Univariate probability distributions». В: *Computational Probability Applications* (2017), с. 133—147.

[43] Erich L Lehmann и George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

[44] Erich Leo Lehmann и др. «Statistical methods based on ranks». В: *Nonparametrics. San Francisco, CA, Holden-Day* 2 (1975).

[45] Faming Liang, Chuanhai Liu и Raymond Carroll. *Advanced Markov chain Monte Carlo methods: learning from past samples*. John Wiley & Sons, 2011.

[46] Thomas Luu. «Fast and accurate parallel computation of quantile functions for random number generation». Дис. ... док. UCL (University College London), 2016.

[47] Bruce D McCullough. «Assessing the reliability of statistical software: Part I». В: *The American Statistician* 52.4 (1998), с. 358—366.

[48] Bruce D McCullough. «Assessing the reliability of statistical software: Part II». В: *The American Statistician* 53.2 (1999), с. 149—159.

[49] Nicholas Metropolis и Stanislaw Ulam. «The monte carlo method». В: *Journal of the American statistical association* 44.247 (1949), с. 335—341.

[50] John F Monahan. «Accuracy in random number generation». В: *Mathematics of Computation* 45.172 (1985), с. 559—568.

[51] John F Monahan. *Numerical methods of statistics*. Cambridge University Press, 2011.

[52] Govind S Mudholkar и Alan D Hutson. «LQ-moments: Analogs of L-moments». В: *Journal of Statistical Planning and Inference* 71.1-2 (1998), с. 191—208.

[53] Frank Nielsen и Vincent Garcia. «Statistical exponential families: A digest with flash cards». В: *arXiv preprint arXiv:0911.4863* (2009).

[54] Peter Occil. *Partially-Sampled Random Numbers for Accurate Sampling of Continuous Distributions*. 2020. URL: https://peteroupc.github.io/exporand.html.

[55] Luigi Pace и Alessandra Salvan. *Principles of statistical inference: from a Neo-Fisherian perspective*. Т. 4. World scientific, 1997.

[56] Xavier Pennec. «Probabilities and statistics on Riemannian manifolds: Basic tools for geometric measurements.» В: *NSIP*. Т. 3. 1999, с. 194—198.

[57] Karsten Prause и др. «The generalized hyperbolic model: Estimation, financial derivatives, and risk measures». Дис. ... док. Citeseer, 1999.

[58] *PySATL*. https://github.com/PySATL.

[59] David Ruppert. «What is kurtosis? An influence function approach». В: *The American Statistician* 41.1 (1987), с. 1—5.

[60] W. A. Stein и др. *Sage Mathematics Software (Version x.y.z)*. The Sage Development Team.

[61] Jacob Schreiber. «Pomegranate: fast and flexible probabilistic modeling in python». В: *Journal of Machine Learning Research* 18.164 (2018), с. 1—6.

[62] *SciLab*. https://www.scilab.org/.

[63] Melvin Dale Springer. *The algebra of random variables*. New York: Wiley, 1979.

[64] György Steinbrecher и William T Shaw. «Quantile mechanics». В: *European journal of applied mathematics* 19.2 (2008), с. 87—112.

[65] Maciej J Swat, Pierre Grenon и Sarala Wimalaratne. «ProbOnto: ontology and knowledge base of probability distributions». В: *Bioinformatics* 32.17 (2016), с. 2719—2721.

[66] Ronald Aaron Thisted. *Elements of statistical computing: Numerical computation*. Routledge, 2017.

[67] G Tsamasphyros и PS Theocaris. «Numerical inversion of Mellin transforms». В: *BIT Numerical Mathematics* 16.3 (1976), с. 313—321.

[68] Pauli Virtanen и др. «SciPy 1.0: fundamental algorithms for scientific computing in Python». В: *Nature methods* 17.3 (2020), с. 261—272.

[69] John Von Neumann и др. «Various techniques used in connection with random digits». В: *John von Neumann, Collected Works* 5 (1963), с. 768—770.

[70] Byron C. Wallace и др. «Closing the Gap between Methodologists and End-Users: R as a Computational Back-End». В: *Journal of Statistical Software* 49.5 (2012), с. 1—15. DOI: 10.18637/jss.v049.i05. URL: https://www.jstatsoft.org/index.php/jss/article/view/v049i05.

[71] Lance A Waller, Bruce W Turnbull и J Michael Hardin. «Obtaining distribution functions by numerical inversion of characteristic functions with applications». В: *The American Statistician* 49.4 (1995), с. 346—350.

[72] Mitchell Watnik. «Early computational statistics». В: *Journal of Computational and Graphical Statistics* 20.4 (2011), с. 811—817.

[73] Herbert Weisberg. *Central tendency and variability*. 83. Sage, 1992.

[74] Robert Charles Williamson и др. «Probabilistic arithmetic». Дис. ... док. University of Queensland Brisbane, 1989.

[75] *Wolfram Mathematica*. https://www.wolfram.com/mathematica/.

[76] Владимир Николаевич Задорожный. «Каскадный метод реализации распределений с тяжелыми хвостами». В: *Омский научный вестник* 2 (140) (2015), с. 222—226.

[77] Владимир Николаевич Задорожный и Олег Иванович Кутузов. «Проблемы генерации случайных величин с фрактальными распределениями». В: *Омский научный вестник* 3 (113) (2012), с. 20—24.

[78] Вильям Феллер. *Введение в теорию вероятностей и ее приложения*. Рипол Классик, 2013.

[79] Наталья Исааковна Чернова. *Математическая статистика*. Новосибирский гос. ун-т, 2007.

[80] Альберт Николаевич Ширяев. *Вероятность*. МЦНМО, 2007.