

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«Московский физико-технический институт  
(национальный исследовательский университет)»**

**УТВЕРЖДЕНО**

**Директор физтех-школы  
прикладной математики и  
информатики**

**А.М. Райгородский**

	<b>Рабочая программа дисциплины (модуля)</b>
<b>по дисциплине:</b>	Введение в анализ данных
<b>по направлению:</b>	Прикладная математика и информатика
<b>профиль подготовки:</b>	Прикладная математика, компьютерные науки и инженерия Физтех-школа Прикладной Математики и Информатики кафедра дискретной математики
<b>курс:</b>	2
<b>квалификация:</b>	бакалавр

Семестр, формы промежуточной аттестации: 4 (весенний) - Дифференцированный зачет

Аудиторных часов: 30 всего, в том числе:

лекции: 30 час.

семинары: 0 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 60 час.

Всего часов: 90, всего зач. ед.: 2

Программу составил: И.Г. Эрлих, канд. физ.-мат. наук

Программа обсуждена на заседании кафедры дискретной математики 04.06.2020

## Аннотация

В данном курсе студенты изучат различные инструменты для анализа данных, работу с данными, визуализацию, познакомятся с различными задачами анализа данных, а также подготовка данных. А также применяют практический материал по теории вероятностей с визуализацией вероятностных объектов и законов на языке Питон.

### 1. Цели и задачи

#### Цель дисциплины

Изучить различные инструменты для анализа данных, работу с данными, визуализацию, применив для практических задач теории вероятностей с визуализацией вероятностных объектов и законов на языке Питон.

#### Задачи дисциплины

- ☐ Познакомить студентов с некоторыми применениями интеллектуального анализа данных и машинного обучения для решения бизнес-задач;
- ☐ научить студентов видеть проблемы, которые могут быть решены с помощью машинного обучения;
- ☐ научить студентов осуществлять постановку задач интеллектуального анализа данных.

### 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-1 Способен применять фундаментальные знания, полученные в области физико-математических и (или) естественных наук и использовать их в профессиональной деятельности	ОПК-1.1 Способен анализировать поставленную задачу, намечать пути ее решения
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать новые научные результаты	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности

### 3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- ☐ Постановки задач интеллектуального анализа данных;
- ☐ популярные алгоритмы интеллектуального анализа данных;
- ☐ современный технический уровень в развитии алгоритмов интеллектуального анализа данных.

уметь:

- ☐ Находить в описании задач из бизнеса задачи для интеллектуального анализа данных;
- ☐ осуществлять математическую постановку задач интеллектуального анализа данных.

владеть:

- ☐ Современными алгоритмами интеллектуального анализа данных;
- ☐ современным инструментарием для промышленного решения задач интеллектуального анализа данных.

#### 4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

##### 4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Обзор задач анализа данных, объекты анализа данных.	6			12
2	Настройка окружения.	6			12
3	Библиотека seaborn, визуализация данных: ядерные оценки, бокс плоты и др.	6			12
4	Вероятностные распределения, их практическое применение, способы генерации псевдослучайных чисел.	6			12
5	Визуализация теории вероятностей, условные вероятности, задачи на графах зависимостей.	6			12
Итого часов		30			60
Подготовка к экзамену		0 час.			
Общая трудоёмкость		90 час., 2 зач.ед.			

##### 4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

###### Семестр: 4 (Весенний)

###### 1. Обзор задач анализа данных, объекты анализа данных.

Примеры прикладных задач: проверка эффективности лекарств, синтез речи и распознавание речи, использование статистических методов в рекомендательных системах, прикладная аналитика.

###### 2. Настройка окружения.

Анаконда, mamba, fish, tmux, cookiecutter. Особенности и расширения Jupyter Notebook.

###### 3. Библиотека seaborn, визуализация данных: ядерные оценки, бокс плоты и др.

Особенности создания датасетов на практике, обработка сырых данных.

###### 4. Вероятностные распределения, их практическое применение, способы генерации псевдослучайных чисел.

Библиотеки numpy, scipy.stats, matplotlib, pandas

###### 5. Визуализация теории вероятностей, условные вероятности, задачи на графах зависимостей.

Практическое применение предельных теоремы теории вероятностей: закон больших чисел, центральная предельная теорема.

## **5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)**

Учебная аудитория, оснащенная компьютером и мультимедийным оборудованием (проектор, звуковая система).

## **6. Перечень рекомендуемой литературы**

### **Основная литература**

1. Наглядная математическая статистика [Текст] : учеб. пособие для вузов / М. Б. Лагутин .— 7-е изд. — М. : БИНОМ. Лаб. знаний, 2019 .— 472 с. : ил. - Библиогр.: с. 456-459. - Предм. указ.: с. 462-466. - ISBN 978-5-00101-105-7 (в пер.) .— Полный текст (Режим доступа : доступ из сети МФТИ).

### **Дополнительная литература**

1. Прикладная математическая статистика [Текст] : для инженеров и научных работников / А. И. Кобзарь .— 2-е изд., испр. / [Научное изд.] .— М. : Физматлит, 2012 .— 816 с. — (Современные методы в математике). - Библиогр.: с. 737-759. - Предм. указ.: с. 806-810. - Имен. указ.: с. 811-813. - 500 экз. - ISBN 978-5-9221-1375-5 (в пер.) .— Полный текст (Доступ из сети МФТИ / Удаленный доступ).

## **7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)**

Не используются

## **8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)**

На занятиях используются мультимедийные технологии, включая демонстрацию презентаций. Для контроля и коррекции знаний, обучающиеся могут использовать компьютерное тестирование.

## **9. Методические указания для обучающихся по освоению дисциплины (модуля)**

Для успешного освоения данной дисциплины студенту необходимо:

- ☐ посещать семинары, при этом конспектирование материалов не является необходимым, поскольку основные материалы хранятся в кафедральной папке в облачном хранилище данных «Яндекс. Диск», к которому предоставлен доступ всем студентам кафедры;
- ☐ выполнять задания, задаваемые преподавателем на семинарах;
- ☐ выполнить итоговое письменное задание по дисциплине, которое вносит основной вклад в изучение дисциплины, а также в итоговую оценку по данному курсу.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

<b>по направлению:</b>	Прикладная математика и информатика
<b>профиль подготовки:</b>	Прикладная математика, компьютерные науки и инженерия Физтех-школа Прикладной Математики и Информатики кафедра дискретной математики
<b>курс:</b>	2
<b>квалификация:</b>	бакалавр
Семестр, формы промежуточной аттестации: 4 (весенний) - Дифференцированный зачет	
<b>Разработчик:</b>	И.Г. Эрлих, канд. физ.-мат. наук

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-1 Способен применять фундаментальные знания, полученные в области физико-математических и (или) естественных наук и использовать их в профессиональной деятельности	ОПК-1.1 Способен анализировать поставленную задачу, намечать пути ее решения
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать новые научные результаты	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Введение в анализ данных» обучающийся должен:

### знать:

- ☐ Постановки задач интеллектуального анализа данных;
- ☐ популярные алгоритмы интеллектуального анализа данных;
- ☐ современный технический уровень в развитии алгоритмов интеллектуального анализа данных.

### уметь:

- ☐ Находить в описании задач из бизнеса задачи для интеллектуального анализа данных;
- ☐ осуществлять математическую постановку задач интеллектуального анализа данных.

### владеть:

- ☐ Современными алгоритмами интеллектуального анализа данных;
- ☐ современным инструментарием для промышленного решения задач интеллектуального анализа данных.

## 3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Примеры заданий приведены в конце программы в виде отдельного файла

## 4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. Назовите основные задачи анализа данных.
2. Что необходимо для построения предсказательной модели?
3. Назовите основные библиотеки на Питон для работы с данными.
4. Какие есть способы первичной визуализации данных?
5. В чем заключается трудность обработки данных с пропусками?
6. Какие основные правила сбора и обработки данных?
7. Где на практике используются основные вероятностные распределения?
8. В чем практическое применение предельных теорем теории вероятностей?

Билет 1:

1. Назовите основные библиотеки на Питон для работы с данными.
2. Какие есть способы первичной визуализации данных?

Билет 2:

1. Какие основные правила сбора и обработки данных?
2. Где на практике используются основные вероятностные распределения?

#### Критерии оценивания

- оценка «отлично (10)» выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений
- оценка «отлично (9)» выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений
- оценка «отлично (8)» выставляется студенту, показавшему всесторонние систематизированные, глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, и правильное обоснование принятых решений
- оценка «хорошо (7)» выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «хорошо (6)» выставляется студенту, если он знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «хорошо (5)» выставляется студенту, если он знает материал, и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;
- оценка «удовлетворительно (4)» выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;
- оценка «удовлетворительно (3)» выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет фрагментарно основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;
- оценка «неудовлетворительно (2)» выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач
- оценка «неудовлетворительно (1)» выставляется студенту, который не знает формулировок основных понятий дисциплины.

#### **5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности**

При проведении дифференцированного зачета, обучающемуся предоставляется 30 минут на подготовку. Опрос обучающегося на зачете не должен превышать двух астрономических часов. Во время проведения дифференцированного зачета, обучающиеся могут пользоваться программой дисциплины, а также справочной литературой и другими материалами.

# Введение в анализ данных

## Домашнее задание 1. Numpy, matplotlib, scipy.stats

### Правила:

- Дедлайн **25 марта 23:59**. После дедлайна работы не принимаются кроме случаев наличия уважительной причины.
- Выполненную работу нужно отправить на почту `mipt.stats@yandex.ru`, указав тему письма "[номер группы] Фамилия Имя - Задание 1". Квадратные скобки обязательны.
- Прислать нужно ноутбук и его pdf-версию (без архивов). Названия файлов должны быть такими: `1.N.ipynb` и `1.N.pdf`, где `N` -- ваш номер из таблицы с оценками. *pdf-версию можно сделать с помощью Ctrl+P. Пожалуйста, посмотрите ее полностью перед отправкой. Если что-то существенное не напечатается в pdf, то баллы могут быть снижены.*
- Решения, размещенные на каких-либо интернет-ресурсах, не принимаются. Кроме того, публикация решения в открытом доступе может быть приравнена к предоставлению возможности списать.
- Для выполнения задания используйте этот ноутбук в качестве основы, ничего не удаляя из него.
- Пропущенные описания принимаемых аргументов дописать на русском.
- Если код будет не понятен проверяющему, оценка может быть снижена.

### Баллы за задание:

Легкая часть (достаточно на "хор"):

- Задача 1.1 -- 3 балла
- Задача 1.2 -- 3 балла
- Задача 2 -- 3 балла

Сложная часть (необходимо на "отл"):

- Задача 1.3 -- 3 балла
- Задача 3.1 -- 3 балла
- Задача 3.2 -- 3 балла
- Задача 3.3 -- 3 балла
- Задача 4 -- 4 балла

Баллы за разные части суммируются отдельно, нормируются впоследствии также отдельно. Иначе говоря, 1 балл за легкую часть может быть не равен 1 баллу за сложную часть.

In [ ]:

```
1 import numpy as np
2 import scipy.stats as sps
3
4 import matplotlib.pyplot as plt
5 import matplotlib.cm as cm
6 from mpl_toolkits.mplot3d import Axes3D
7 import ipywidgets as widgets
8
9 import typing
10
11 %matplotlib inline
```



## Легкая часть: генерация

В этой части другие библиотеки использовать запрещено. Шаблоны кода ниже менять нельзя.

### Задача 1

Имеется симметричная монета. Напишите функцию генерации независимых случайных величин из нормального и экспоненциального распределений с заданными параметрами.

In [ ]:

```
1 ▾ # Эта ячейка -- единственная в задаче 1, в которой нужно использовать
2   # библиотечную функцию для генерации случайных чисел.
3   # В других ячейках данной задачи используйте функцию coin.
4
5   # симметричная монета
6   coin = sps.<ваш код>
```

Проверьте работоспособность функции, сгенерировав 10 бросков симметричной монеты.

In [ ]:

```
1 coin(size=10)
```

**Часть 1.** Напишите сначала функцию генерации случайных величин из равномерного распределения на отрезке  $[0, 1]$  с заданной точностью. Это можно сделать, записав случайную величину  $\xi \sim U[0, 1]$  в двоичной системе счисления  $\xi = 0, \xi_1 \xi_2 \xi_3 \dots$ . Тогда  $\xi_i \sim \text{Bern}(1/2)$  и независимы в совокупности. Приближение заключается в том, что вместо генерации бесконечного количества  $\xi_i$  мы полагаем  $\xi = 0, \xi_1 \xi_2 \xi_3 \dots \xi_n$ .

Нужно реализовать функцию так, чтобы она могла принимать на вход в качестве параметра `size` как число, так и объект `tuple` любой размерности, и возвращать объект `numpy.array` соответствующей размерности. Например, если `size=(10, 1, 5)`, то функция должна вернуть объект размера  $10 \times 1 \times 5$ . Кроме того, функцию `coin` можно вызвать только один раз, и, конечно же, не использовать какие-либо циклы. Аргумент `precision` отвечает за число  $n$ .

In [ ]:

```
1 ▾ def uniform(size=1, precision=30):
2   return # В одну строчку не выходя за границы? ;)
```

Для  $U[0, 1]$  сгенерируйте 200 независимых случайных величин, постройте график плотности на отрезке  $[-0.25, 1.25]$ , а также гистограмму по сгенерированным случайным величинам.

In [ ]:

```
1 size = 200
2 ▾ grid = <функция из numpy для создания равномерной сетки
3       от -0.25 до 1.25 на 500 точек>
4 sample = <Сгенерируйте size случайных величин точности 50>
5
6 # Отрисовка графика
7 plt.<определите график размера 10 на 4>
8
9 # отображаем значения случайных величин полупрозрачными точками
10 ▾ plt.<функция отрисовки точек>(
11     sample,
12     np.zeros(size),
13     <прозрачность точки равна 0.4>,
14     <подпись точек в легенде к графику>
15 )
16
17 # по точкам строим нормированную полупрозрачную гистограмму
18 ▾ plt.<функция отрисовки гистограммы>(
19     sample,
20     <10 столбиков>,
21     <нормировка столбиков>,
22     <прозрачность столбиков равна 0.4>,
23     <оранжевый цвет столбиков>
24 )
25
26 # рисуем график плотности
27 ▾ plt.<функция отрисовки линии>(
28     grid,
29     <Посчитайте плотность в точках grid, используя sps.uniform.pdf>,
30     <красный цвет линии>,
31     <толщина линии равна 3>,
32     <подпись линии в легенде к графику>
33 )
34 plt.<легенда>
35 plt.<сетка>(ls=':')
36 plt.show()
```

Исследуйте, как меняются значения случайных величин в зависимости от precision .

In [ ]:

```
1 size = 100
2
3 plt.<определите график размера 15 на 3>
4
5 ▾ for i, precision in enumerate([1, 2, 3, 5, 10, 30]):
6     plt.<определите подграфик>(3, 2, i + 1)
7     ▾ plt.<функция отрисовки точек>(
8         <Сгенерируйте выборку размера size точности precision>,
9         np.zeros(size),
10         <прозрачность точки равна 0.4>
11     )
12     plt.yticks([])
13     if i < 4: plt.xticks([])
14
15 plt.show()
```

## Вывод:

<...>

**Часть 2.** Напишите функцию генерации случайных величин в количестве `size` штук (как и раньше, тут может быть `tuple`) из распределения  $\mathcal{N}(loc, scale^2)$  с помощью преобразования Бокса-Мюллера, которое заключается в следующем. Пусть  $\xi$  и  $\eta$  -- независимые случайные величины, равномерно распределенные на  $(0, 1]$ . Тогда случайные величины  $X = \cos(2\pi\xi)\sqrt{-2\ln\eta}$ ,  $Y = \sin(2\pi\xi)\sqrt{-2\ln\eta}$  являются независимыми нормальными  $\mathcal{N}(0, 1)$ .

Реализация должна быть без циклов. Желательно использовать как можно меньше бросков монеты.

In [ ]:

```
1 ▾ def normal(size=1, loc=0, scale=1, precision=30):  
2     <...>
```

Для  $\mathcal{N}(0, 1)$  сгенерируйте 200 независимых случайных величин, постройте график плотности на отрезке  $[-3, 3]$ , а также гистограмму по сгенерированным случайным величинам.

In [ ]:

```
1     <...>
```

## Сложная часть: генерация

**Часть 3.** Вы уже научились генерировать выборку из равномерного распределения. Напишите функцию генерации выборки из экспоненциального распределения, используя из теории вероятностей:

*Если  $\xi$  --- случайная величина, имеющая абсолютно непрерывное распределение, и  $F$  --- ее функция распределения, то случайная величина  $F(\xi)$  имеет равномерное распределение на  $[0, 1]$ .*

Какое преобразование над равномерной случайной величиной необходимо совершить?

<...>

Для получения полного балла реализация должна быть без циклов, а параметр `size` может быть типа `tuple`.

In [ ]:

```
1 ▾ def expon(size=1, lambd=1, precision=30):  
2     return <...>
```

Для  $Exp(1)$  сгенерируйте выборку размера 100 и постройте график плотности этого распределения на отрезке  $[-0.5, 5]$ .

In [ ]:

```
1     <...>
```

## Вывод по задаче:

<...>

## Легкая часть: матричное умножение

### Задача 2

Напишите функцию, реализующую матричное умножение. При вычислении разрешается создавать объекты размерности три. Запрещается пользоваться функциями, реализующими матричное умножение (`numpy.dot`, операция `@`, операция умножения в классе `numpy.matrix`). Разрешено пользоваться только простыми векторно-арифметическими операциями над `numpy.array`, а также преобразованиями осей. Авторское решение занимает одну строчку.

In [ ]:

```
1 ▾ def matrix_multiplication(A, B):  
2     return <...>
```

Проверьте правильность реализации на случайных матрицах. Должен получиться ноль.

In [ ]:

```
1 A = sps.uniform.rvs(size=(10, 20))  
2 B = sps.uniform.rvs(size=(20, 30))  
3 np.abs(matrix_multiplication(A, B) - A @ B).sum()
```

На основе опыта: вот в таком стиле многие из вас присылали бы нам свои работы, если не стали бы делать это задание :)

In [ ]:

```
1 ▾ def stupid_matrix_multiplication(A, B):  
2     C = [[0 for j in range(len(B[0]))] for i in range(len(A))]  
3     for i in range(len(A)):  
4         for j in range(len(B[0])):  
5             for k in range(len(B)):  
6                 C[i][j] += A[i][k] * B[k][j]  
7     return C
```

Проверьте, насколько быстрее работает ваш код по сравнению с неэффективной реализацией `stupid_matrix_multiplication`. Эффективный код должен работать почти в 200 раз быстрее. Для примера посмотрите также, насколько быстрее работают встроенные `numpy`-функции.

In [ ]:

```
1 A = sps.uniform.rvs(size=(400, 200))  
2 B = sps.uniform.rvs(size=(200, 300))  
3  
4 %time C1 = matrix_multiplication(A, B)  
5 %time C2 = A @ B # python 3.5  
6 %time C3 = np.matrix(A) * np.matrix(B)  
7 %time C4 = stupid_matrix_multiplication(A, B)  
8 %time C5 = np.einsum('ij,jk->ik', A, B)
```

Ниже для примера приведена полная реализация функции. Вас мы, конечно, не будем требовать проверять входные данные на корректность, но документации к функциям нужно писать.

In [ ]:

```
1  def matrix_multiplication(A, B):
2      '''Возвращает матрицу, которая является результатом
3      матричного умножения матриц A и B.
4
5      ...
6
7      # Если A или B имеют другой тип, нужно выполнить преобразование типов
8      A = np.array(A)
9      B = np.array(B)
10
11     # Проверка данных входных данных на корректность
12     assert A.ndim == 2 and B.ndim == 2, 'Размер матриц не равен 2'
13     assert A.shape[1] == B.shape[0], \
14         ('Матрицы размерностей {} и {} неперемножаемы'.format(A.shape, B.shape))
15
16     C = <...>
17
18     return C
```

## Сложная часть: броуновское движение

### Задача 3

**Познавательная часть задачи** (не пригодится для решения задачи)

Абсолютное значение скорости движения частиц идеального газа, находящегося в состоянии ТД-равновесия, есть случайная величина, имеющая распределение Максвелла и зависящая только от одного термодинамического параметра — температуры  $T$ .

В общем случае плотность вероятности распределения Максвелла для  $n$ -мерного пространства имеет вид:

$$p(v) = C e^{-\frac{mv^2}{2kT}} v^{n-1},$$

где  $v \in [0, +\infty)$ , а константа  $C$  находится из условия нормировки  $\int_0^{+\infty} p(v) dv = 1$ .

Физический смысл этой функции таков: вероятность того, что скорость частицы входит в промежуток  $[v_0, v_0 + dv]$ , приближённо равна  $p(v_0)dv$  при достаточно малом  $dv$ . Тут надо оговориться, что математически корректное утверждение таково:

$$\lim_{dv \rightarrow 0} \frac{P\{v \mid v \in [v_0, v_0 + dv]\}}{dv} = p(v_0).$$

Поскольку это распределение не ограничено справа, определённая доля частиц среды приобретает настолько высокие скорости, что при столкновении с макрообъектом может происходить заметное отклонение как траектории, так и скорости его движения.

Мы предполагаем идеальность газа, поэтому компоненты вектора скорости частиц среды  $v_i$  можно считать независимыми нормально распределёнными случайными величинами, т.е.

$$v_i \sim \mathcal{N}(0, s^2),$$

где  $s$  зависит от температуры и массы частиц и одинаково для всех направлений движения.

При столкновении макрообъекта с частицами среды происходит перераспределение импульса в соответствии с законами сохранения энергии и импульса, но в силу большого числа подобных событий за единицу времени, моделировать их напрямую достаточно затруднительно. Поэтому для выполнения этого ноутбука сделаем следующие предположения:

- Приращение компоненты координаты броуновской частицы за фиксированный промежуток времени (или за шаг)  $\Delta t$  имеет вид  $\Delta x_i \sim \mathcal{N}(0, \sigma^2)$ .
- $\sigma$  является конкретным числом, зависящим как от  $\Delta t$ , так и от параметров броуновской частицы и среды.
- При этом  $\sigma$  не зависит ни от координат, ни от текущего вектора скорости броуновской частицы.

Если говорить формальным языком, в этом ноутбуке мы будем моделировать [Винеровский случайный процесс](#)  
[\(https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%BD%D0%B5%D1%80%D0%BE%D0%B2%D1%81%D0%\)](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%BD%D0%B5%D1%80%D0%BE%D0%B2%D1%81%D0%)  
с фиксированным шагом.

## Задание

### 1. Разработать функцию симуляции броуновского движения

Функция должна вычислять приращение координаты частицы на каждом шаге как  $\Delta x_{ijk} \sim \mathcal{N}(0, \sigma^2) \forall i, j, k$ , где  $i$  — номер частицы,  $j$  — номер координаты, а  $k$  — номер шага. Функция принимает в качестве аргументов:

- Параметр  $\sigma$ ;
- Количество последовательных изменений координат (шагов), приходящихся на один процесс;
- Число процессов для генерации (количество различных частиц);
- Количество пространственных измерений для генерации процесса.

Возвращаемое значение:

- 3-х мерный массив `result`, где `result[i,j,k]` — значение  $j$ -й координаты  $i$ -й частицы на  $k$ -м шаге.

### Общее требование

- Считать, что все частицы в начальный момент времени находятся в начале координат.

## Что нужно сделать

- Реализовать функцию для произвольной размерности, не используя циклы.
- Дописать проверки типов для остальных аргументов.

Обратите внимание на использование аннотаций для типов аргументов и возвращаемого значения функции. В новых версиях Питона подобные возможности синтаксиса используются в качестве подсказок для программистов и статических анализаторов кода, и никакой дополнительной функциональности не добавляют.

Например, `typing.Union[int, float]` означает "или `int`, или `float`".

### Что может оказаться полезным

- Генерация нормальной выборки: `scipy.stats.norm`. [Ссылка](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html) (<https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html>).
- Кумулятивная сумма: метод `cumsum` у `np.ndarray`. [Ссылка](https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.cumsum.html) (<https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.cumsum.html>).

In [ ]:

```
1  def generate_brownian(sigma: typing.Union[int, float] = 1,
2                          *,
3                          n_proc: int = 10,
4                          n_dims: int = 2,
5                          n_steps: int = 100) -> np.ndarray:
6      """
7      :param sigma:    стандартное отклонение нормального распределения,
8                      генерирующего пошаговые смещения координат
9      :param n_proc:    <ДОПИСАТЬ>
10     :param n_dims:    <ДОПИСАТЬ>
11     :param n_steps:   <ДОПИСАТЬ>
12
13     :return:          np.ndarray размера (n_proc, n_dims, n_steps), содержащий
14                      на позиции [i,j,k] значение j-й координаты i-й частицы
15                      на k-м шаге.
16     """
17     if not np.issubdtype(type(sigma), np.number):
18         raise TypeError("Параметр 'sigma' должен быть числом")
19     # <ДОПИСАТЬ ПРОВЕРКИ ТИПОВ>
20
21     return <...>
```

Символ `*` в заголовке означает, что все аргументы, объявленные после него, необходимо определять только по имени.

Например,

```
generate_brownian(323, 3)           # Ошибка
generate_brownian(323, n_steps=3)   # OK
```

При проверке типов остальных аргументов, по аналогии с `np.number`, можно использовать `np.integer`. Конструкция `np.issubdtype(type(param), np.number)` используется по причине того, что стандартная питоновская проверка `isinstance(sigma, (int, float))` не будет работать для `numpy`-чисел `int64`, `int32`, `float64` и т.д.

In [ ]:

```
1  brownian_2d = generate_brownian(2, n_steps=12000, n_proc=500, n_dims=2)
2  assert brownian_2d.shape == (500, 2, 12000)
```

## 2. Визуализируйте траектории для 9-ти первых броуновских частиц

### Что нужно сделать

- Нарисовать 2D-графики для `brownian_2d`.

- Нарисовать 3D-графики для `brownian_3d = generate_brownian(2, n_steps=12000, n_proc=500, n_dims=3)`.

### Общие требования

- Установить соотношение масштабов осей, равное 1, для каждого из подграфиков.

### Что может оказаться полезным

- [Тьюториал \(https://matplotlib.org/devdocs/gallery/subplots\\_axes\\_and\\_figures/subplots\\_demo.html\)](https://matplotlib.org/devdocs/gallery/subplots_axes_and_figures/subplots_demo.html) по построению нескольких графиков на одной странице.
- Метод `plot` у `AxesSubplot` (переменная `ax` в цикле ниже).
- Метод `set_aspect` у `AxesSubplot`.

In [ ]:

```
1 fig, axes = plt.subplots(3, 3, figsize=(18, 10))
2 fig.suptitle('Траектории броуновского движения', fontsize=20)
3
4 ▼ for ax, (xs, ys) in zip(axes.flat, brownian_2d):
5     <...>
6     pass
```

### 3. Постройте график среднего расстояния частицы от начала координат в зависимости от времени (шага)

- Постройте для `n_dims` от 1 до 5 включительно.
- Кривые должны быть отрисованы на одном графике. Каждая кривая должна иметь легенду.
- Для графиков подписи к осям обязательны.

### Вопросы

- Как вы думаете, какой функцией может описываться данная зависимость?
- Сильно ли её вид зависит от размерности пространства?
- Можно ли её линеаризовать? Если да, нарисуйте график с такими же требованиями.

In [ ]:

```
1 plt.figure(figsize=(12, 6))
2
3 ▼ for n_dims in range(1, 6):
4 ▼     plt.plot(
5         <...>
6         label=f'Размерность: {n_dims}'
7     )
8
9 plt.ylabel('Ср. раст. частицы от нач. координат')
10 plt.xlabel('Шаг')
11 plt.legend(loc='best')
12 plt.tight_layout()
13 plt.show()
```

### Сложная часть: визуализация распределений



## Задача 4

В этой задаче вам нужно исследовать свойства дискретных распределений и абсолютно непрерывных распределений.

Для перечисленных ниже распределений нужно

- 1) На основе графиков дискретной плотности (функции массы) для различных параметров пояснить, за что отвечает каждый параметр.
- 2) Сгенерировать набор независимых случайных величин из этого распределения и построить по ним гистограмму.
- 3) Сделать выводы о свойствах каждого из распределений.

Распределения:

- Бернулли
- Биномиальное
- Равномерное
- Геометрическое

Для выполнения данного задания можно использовать код с лекции.

In [ ]:

1	<...>
---	-------