Оглавление

[Основные термины и определения 2](#_Toc94915333)

[Введение 4](#_Toc94915334)

[Обзор основных видов стохастического динамического программирования 5](#_Toc94915335)

[Объяснения выбора Jupiter Notebook в качестве среды разработки проекта 6](#_Toc94915336)

[Что из себя будет представлять проект 7](#_Toc94915337)

[Описание функциональных требований 8](#_Toc94915338)

[Описание нефункциональных требований 8](#_Toc94915339)

[Список литературы 9](#_Toc94915340)

# **Основные термины и определения**

1. *Случайная величина –* переменная, значение которой не известно заранее, но для которой известно множество возможных значений и вероятностное распределение по этому множеству.

Пример 1: подбрасывание монетки.

Подбрасывается правильная монетка. Пусть Х = 0, если выпал орёл. X = 1, если выпала решка. Тогда X – случайная величина, принимающая значение 0 с вероятностью ½ и значение 1 с вероятностью ½.

Пример 2: бросание кубика.

Кидается правильная игральная кость, каждая грань может выпасть с равной вероятностью. Пусть Х – значение на кубике. Тогда Х – случайная величина, принимающая значения из множества {1, 2, 3, 4, 5, 6}, причём каждое из своих возможных значений Х принимает с вероятностью 1/6.

Пример 3: равномерное распределение.

Случайна величина Х – произвольно выбранная точка на отрезке [a, b] вещественной координатной прямой.

1. *Математическое ожидание случайной величины –* среднее значение случайной величины. Говоря простым языком, если представить, что случайная величина – это результат некоторого эксперимента, то мат. ожидание - это значение, которое мы будем получать в среднем, если проведём множество (кол-во, стремящееся к бесконечности) независимых экспериментов.

Обозначение: E(x) – мат. ожидание случайной величины x.

1. *Перекрещивающиеся задачи* – задачи, решающиеся последовательно, причём для решения каждой задачи используются результаты, полученные при решении некоторых или всех предыдущих задач.

Пример: вычисление чисел Фибоначчи.

F1 = 1, F2 = 1

F3 = F1 + F2

F4 = F2 + F3

…

Fn = Fn - 2 + Fn - 1

1. *Динамическое программирование* – метод решения задачи, путём разбиения её на перекрещивающиеся подзадачи.
2. *Стохастическое программирование* – класс методов решения задач в условиях неполной информации (некоторые параметры задачи являются случайными величинами).
3. *Стохастическое динамическое программирование* – стохастический способ решения задачи путём динамического программирования (стохастический и одновременно динамический метод).
4. *GitHub* – популярная платформа для выкладывания кода в интернет в публичном или приватном доступе.

# Введение

Потребность в стохастическом программировании может возникнуть при работе в самых разных областях. Проблема неопределённости параметров часто встречается в экономических, финансовых, научных и инженерных задачах. Чрезвычайно важно обращаться именно к стохастическому программированию при работе с решениями, на которые влияет неопределённость. Примечательно, что большинство проблем принятия решений относятся к этому типу.

Технически стохастические программы гораздо сложнее, чем соответствующие детерминированные программы. Следовательно, причины для обращения на практике к стохастическим моделям должны быть очень вескими. На самом деле, такие причины действительно есть: альтернативные детерминистические подходы не способны найти наилучшее решение, просто в силу своей природы: они не могут работать в условиях какой-либо неопределённости. Можно, конечно, найти оптимальное решение для всех возможных значений параметров. Но это не даст нам ответ на вопрос, как действовать, если мы пока не знаем точного значения параметров, а принять решение уже надо. Более того, оптимальное для конкретного набора значений решение часто очень плохо при всех других возможных раскладах. Замечательные примеры таких ситуаций можно найти, например, в книге P. Kall и S. Wallace «Stochastic Programming».

Однако стохастическое программирование – очень объёмная тема, и чтобы уместить работу в рамки студенческого проекта на 2-ом курсе, было решено сузить тему и рассмотреть только стохастическое динамическое программирование, как один из самых объёмных и интересных его подвидов.

**Целью данного проекта** является знакомство со стохастическим динамическим программированием, анализ алгоритмов, применяемых в этой области и их реализация.

**Задачи проекта:**

1. Изучить возможные способы решения задач методом стохастического динамического программирования.
2. Решить этими способами ряд учебных и, если получится, реальных задач на языке программирования Python.
3. Проанализировать эффективность различных подходов.
4. Составить пособие с примерами решения задач методом стохастического динамического программирования.

# Обзор основных видов стохастического динамического программирования

В работе будут рассмотрены 2 основных приёма работы со стохастической неопределённостью.

1. Самая простая идея при работе с неопределенностью – это воспользоваться таким мощным инструментом из теории вероятностей как мат. ожидание. Действительно, если в качестве значения каждого неопределённого параметра подставить его мат. ожидание, т. е. усреднённое значение, то можно ожидать, что реальные числа не сильно отклоняться от среднего значения и найденное оптимальное решение будет весьма и весьма хорошим для реальных чисел. Тогда задача решается обычным детерминированным динамическим алгоритмом, на вход которому подаются мат. ожидания.
2. Другой способ – потребовать, чтобы величина, которую в задаче требуется максимизировать (например, прибыль), была не меньше заданного значения с очень высокой вероятностью (с вероятностью не меньше, чем какое-то близкое к единице число, например, 0,99). Аналогичное условие ставится, если требуется минимизировать рассматриваемую величину.

К примеру, пусть в задаче требуется максимизировать значение переменной X и мы хотим, чтобы X превысило 100 с вероятностью не меньше, чем 0,99. Тогда имеет смысл выдвинуть требование:

P(X > 100) >= 0,99.

Данный метод намного более труднореализуем, но и намного более надёжен. Алгоритмы с использованием мат. ожиданий ориентируются на средние значения, и, по большому счёту, не дают никаких гарантий, ведь мы не учитываем, с какой вероятностью и насколько реальные значения могут отклоняться от прогнозируемых. При использовании же второго способа мы получаем не абсолютную, но твёрдую уверенность в «надёжности» нашего решения. Чтобы окончательно понять, к каким последствиям может привести предложенное алгоритмом решение, можно также определить наихудший из возможных исходов.

# Объяснения выбора Jupiter Notebook в качестве среды разработки проекта

Проект выполняется в среде разработки Jupiter Notebook – инструменте для создание аналитических отчётов, а также написания лабораторных, исследовательских или проектных работ в области компьютерных наук, обладающем широким набором возможностей для визуализации данных. Данная платформа была выбрана для работы над проектом, поскольку позволяет сделать из самого проекта и отчёта к нему единое целое: в одном файле формата ipynb могут содержаться и фрагменты кода, и вывод результата их работы, и текстовые заметки. Код пишется на языке Python, пакет Anaconda представляет большое количество специальных библиотек, облегчающий работу с данными и сложными математическими операциями. Для создания текстовых заметок в Notebook используется язык разметки Markdown, основанный на системе компьютерной вёрстки TeX, и все основные команды аналогичны таковым в LaTeX. Таким образом, Markdown является удобным инструментом для набора текста, содержащего большое количество математических символов. Также Jupiter Notebook предоставляет возможность набирать текст непосредственно на html. Иногда удобно писать, делая вставки html в Markdown – это расширяет спектр возможностей, хотя и понижает читаемость исходного кода заметки. После завершения работы с файлом, он может быть переведён в формат pdf. Эта система представляется очень удобной для формата работы, когда решаются не связанные друг с другом и не очень большие по объёму задачи. Каждую задачу предваряет её условие, обсуждение решения, построение математической модели, если нужно. После каждой задачи будут представлены результаты работы программы и какие-то выводы или замечания, если это будет релевантно. Также весь отчёт по программному проекту будет распределён между фрагментами кода. И код, и текст будут представлены довольно небольшими фрагментами, чередующимися между собой. Отсутствие большого количества подряд идущих текста или кода должно упростить чтение работы.

# Что из себя будет представлять проект

Проект будет представлять из себя папку № 1 с файлами формата ipynb, набранными в Jupiter Notebook, и папку № 2 с теми же файлами, но конвертированными в pdf.

Как уже было сказано, все программы реализуются на языке Python.

Помимо материалов самого проекта (объяснения принципов работы, математических моделей, постановки задач, предложения и доказательства гипотез, самого кода, входных и выходных данных, а также выводов), в папках будет файл (или файлы), содержащий (содержащие) введение, цель, задачи, список литературы и некоторую другую документацию.

Итоговым отчётом по проекту будет являться либо вся папка № 2 целиком с приложенными титульным листом и оглавлением, либо, если папка № 2 получится слишком объёмной, то будет создан отдельный файл, содержащий выжимку из материалов папки № 2.

Папки № 1 и № 2 вместе с дополнительными материалами (README.md, титульный лист, оглавление, возможно итоговый отчёт и презентация о проекте) будут собраны в одном репозитории и выложены на GitHub в открытом доступе.

# Описание функциональных требований

1. Планируется, что программы будут отрабатывать и выдавать результаты в течение не более, чем 1-2 секунд. Но при решении очень больших реальных задач допускается время работа вплоть до нескольких дней.
2. Для каждой решённой задачи планируется представить входные и выходные данные. Программы будут написаны так, чтобы выходные данные выводились в виде связного текста или таблицы. Добиться вывода данных в виде таблицы в Jupiter Notebook можно, например, с помощью библиотеки pandas.
3. Программы должны быть снабжены обильными и подробными комментариями, чтобы упростить понимание принципов их (программ) работы.

# Описание нефункциональных требований

1. Для того, чтобы была возможность запустить программы из папки № 1, на компьютере пользователя должен быть установлен Jupiter Notebook версии 5.7.0 и выше, а также Anaconda 3 2020.02 или выше (теоретически всё может сработать и без Anaconda, но гарантии нет). При этом для просмотра любых файлов проекта никакого специального программного обеспечения не требуется (если просмотр осуществляется на платформе GitHub).
2. Пользователь (читатель пособия) должен иметь опыт чтения математической литературы, знать синтаксис язык Python, уметь читать чужой код и иметь хотя бы базовые знания по дискретной математике, теории вероятностей, алгоритмам и структурам данных.

# Список литературы

1. Хэмди А. Таха «Исследование операций»
2. P. Kall and S. Wallace «Stochastic Programming»
3. В. Д. Власенко «Динамическое и стохастическое программирование»
4. В. Е. Гмурман «Теория вероятностей и математическая статистика»