**4. Мои результаты в задаче оптимизации портфеля**

В главе 3 поставлены три задачи P1+P2 и P3, причем P1+P2 является линейной, а P3 — выпуклой, и утверждается, что при условии гауссовости распределения , решения задач P1+P2 и P3 будут совпадать. Напишем для задач (P1+P2) программу на языке С, которая будет находить решение задачи линейного программирования (P1+P2) симплекс-методом с помощью библиотеки glpk-5.0, и сравним полученные нами результаты с результатами в Таблицах 5 и 6. Кроме того, применим теорему Каруша-Куна-Таккера к выпуклой задаче P3, методом множителей Лагранжа найдем оптимальное значение вектора долей соответствующих инструментов , убедимся, что оптимальное значение действительно единственно и сравним полученные **,** , с результатами из Таблицы 3.

**4.1 Реализация симплекс-метода glpk-5.0 для задачи P1+P2**

Мы предполагаем, что вектор , соответствующий ценам трех исследуемых финансовых инструментов, имеет многомерное нормальное распределение с вектором математических ожиданий и матрицей ковариаций , которые заданы в Таблицах 1 и 2 соответственно.

Изображение выглядит как стол

Автоматически созданное описание

Наша цель — минимизировать и портфеля, что, согласно Теореме 1, можно сделать путем минимизации функции:

Как уже говорилось выше, интеграл в определении (4) для может быть приближен несколькими способами. Например, это может быть сделано путем сэмлирования вектора в соответствии с плотностью . Если сэмлирование сгенерировало нам набор векторов , то соответствующая аппроксимация для такова:

В терминах вспомогательных действительных величин , эта задача эквивалентна минимизации линейного выражения:

при условии выполнении линейных ограничений:

Сэмплирование реализаций вектора из многомерного нормального с вектором математических ожиданий и матрицей ковариаций реализовано на языке Python3 в файле generatey1y2y3.py (число передается при запуске этой программы в качестве аргумента командной строки), в результате чего полученная выборка записывается в файл Generatedy1y2y3.txt.

Изображение выглядит как текст

Автоматически созданное описание

Симплекс-метод нахождения оптимального портфеля находится в файле ProblemsP1P2Glpk.cpp, но для того, чтобы этот файл запустился, нужно скачать библиотеку glpk-5.0. Для этого нужно зайти на сайт <https://www.gnu.org/software/glpk/> и нажать в верхнем горизонтальном меню Downloading, потом на открывшейся странице под заголовком Downloading GLPK найти ссылку <http://ftp.gnu.org/gnu/glpk/> и по ней скачать архив [glpk-5.0.tar.gz](http://ftp.gnu.org/gnu/glpk/glpk-5.0.tar.gz). Потом его надо разархивировать в какую-нибудь папку, в этой папке зайти в подпапку gplk-5.0, там будет файл INSTALL, и там написано, что сначало надо в командной строки из этой подпапки glpk-5.0 запустить команду ./configure, потом make, потом make install. После этого библиотека glpk скачана, и ее можно включать в любой cpp-файл командой #include<glpk.h>. Посмотрим на программу ProblemsP1P2Glpk.cpp:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Также написан скрипт zapusk.sh, который принимает на вход и в качестве аргумента командной строки, сам запускает python3-файл для генерации выборки векторов размера , с этой выборкой запускает программу ProblemsP1P2Glpk.cpp и получает ответ: значения

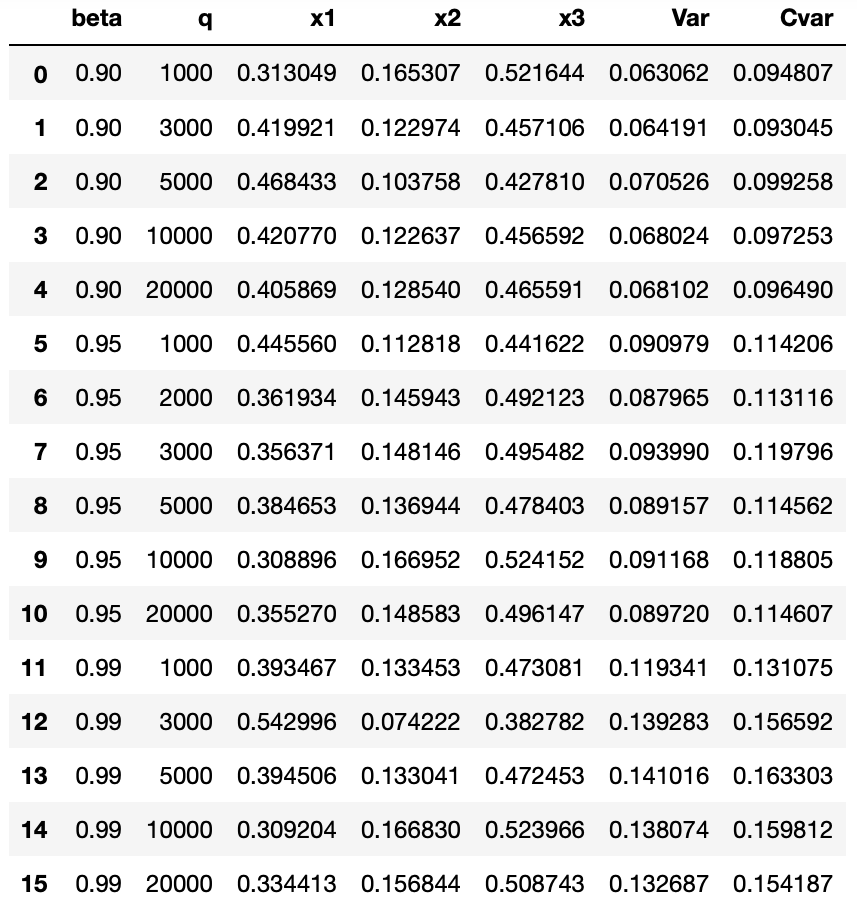
Например, запустим программу с параметрами :

Изображение выглядит как текст

Автоматически созданное описание

Получили значения, очень похожие на значения Таблиц 5 и 6 для соответствующих .

Выполним запуск для и , занесем результаты в таблицу:

**

Видим, что получились результаты, очень близкие к тому, что получили авторы статьи в Таблицах 5 и 6.

**4.2 Реализация метода множителей Лагранжа для задачи P3**

Требуется решить выпуклую задачу P3 на минимум:

При условии ограничений

Для этого применяем метод множителей Лагранжа и получаем единственную допустимую точку: [0.45423611],[0.11471805],[0.43104584], как и в таблице 3. Метод множителей Лагранжа проведен вручную на листочке, вычисления по полученным формулам проведены в Python3 в файле ProblemP3Kursovaya.ipynb.

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание