TOMSK POLYTECHNIC UNIVERSITY
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Инженерная школа информационных технологий и робототехники
Направление подготовки: Прикладная математика и информатика
Отделение информационных технологий

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Решение задачи квадратичного программирования на основе операторов-проекторов

УДК 004.021-047.84:519.853.32:517.98

Студент

Группа	ФИО	Подпись	Дата
8Б51	Кружков Денис Сергеевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Вылегжанин Олег Николаевич	к.х.н.		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Подопригора Игнат Валерьевич	к.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент ООТД	Мезенцева Ирина Леонидовна			

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Шевелев Геннадий Ефимович	к.ф.-м.н., доцент		

Томск – 2019 г.

ЗАПЛАНИРОВАННЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ООП

Код результатов	Результат обучения (выпускник должен быть готов)	Требования ФГОС, Критерии АИОР
Профессиональные компетенции		
Р1	Применять *глубокие математические и профессиональные знания* для решения задач научно-исследовательской, проектной, производственной и технологической деятельности в области системного и прикладного программирования.	Требования ФГОС (ОК-11, 12, ПК-3, 10), Критерий 5 АИОР (п. 5.2.1), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI.* Требования профессиональных стандартов Ассоциации предприятий компьютерных и информационных технологий Требования работодателей: ФГУП «РФЯЦ-ВНИИТФ им. академика Е.И. Забабахина», ООО НАЦ «Недра», ИХН СО РАН
Р2	Умение использовать знания по естественнонаучным дисциплинам при определении задач математического моделирования объектов и явлений в различных предметных областях	Требования ФГОС (ПК-3,9) Критерий 5 АИОР (п.5.2.3), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI.* Требования работодателей: ФГУП «РФЯЦ-ВНИИТФ им. академика Е.И. Забабахина», ООО «НАПО им. В.П.
Р3	Демонстрировать понимание сущности и значения информации в развитии современного общества, владение основными методами, способами и средствами получения, хранения, переработки информации; использование для решения коммуникативных задач современных технических средств и информационных технологий	Требования ФГОС (ОК-5, 11, 12,14,15, ПК-2, 6), Критерий 5 АИОР (п. 5.2.2), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI* Требования профессиональных стандартов Ассоциации предприятий компьютерных и информационных технологий Требования работодателей: Контек, ОАО «Газпром переработка», ООО Нижневартовскэнергонефть». Требования ФГОС (ОК-5, 11,

Р4	Выполнять *инновационные* проекты с применением *глубоких профессиональных* знаний и *эффективных* методов проектирования для достижения *новых* результатов, обеспечивающих *конкурентные преимущества* в условиях экономических, экологических, социальных и других ограничений.	Требования ФГОС (ОК-14, ПК- 7, 9,14), Критерий 5 АИОР (п. 5.2.4), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI* Требования профессиональных стандартов Ассоциации предприятий компьютерных и информационных технологий. Требования работодателей: Контек, ОАО «Газпром переработка», ИХН СО РАН.
Р5	*Демонстрировать знание* о формах организации образовательной и научной деятельности в высших учебных заведениях, *иметь навыки* преподавательской работы.	Требования ФГОС (ОК-1, 10, 16, ПК-1, 14, 15), Критерий 5 АИОР (п. 5.2.1), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI*
Р6	Способность осуществлять организационно-управленческую и социально-ориентированную деятельность с соблюдением профессиональной этики	Требования ФГОС (ОК-5,13,16, ПК-11-13,16) Критерий 5 АИОР (п.5.2.12-13) согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI*
Универсальные компетенции		
Р7	*Активно* владеть *иностранным языком* на уровне, позволяющем работать в интернациональной среде, включая разработку документации и представление результатов инновационной деятельности. Толерантность в восприятии социальных и культурных различий.	Требования ФГОС (ОК-2, 3,4, 7, ПК-8). Критерий 5 АИОР (п. 5.2.11), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI* и Требования профессиональных стандартов Ассоциации предприятий компьютерных и информационных технологий

Р8	Эффективно работать индивидуально, в качестве *члена и руководителя группы*, состоящей из специалистов различных направлений и квалификаций, демонстрировать ответственность за результаты работы и готовность *следовать корпоративной культуре* организации	Требования ФГОС (ОК-1,4, 6, ПК-8,11,12), Критерий 5 АИОР (пп. 5.2.9,5.2.13), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI и* Требования профессиональных стандартов Ассоциации предприятий компьютерных и информационных технологий. Требования работодателей: Контек, ОАО «Газпром переработка», ООО Нижневартовскэнергонефть».
Р9	*Самостоятельно учиться* и непрерывно *повышать квалификацию* в течение всего периода профессиональной деятельности. Способность к интеллектуальному, культурному, нравственному и профессиональному саморазвитию.	Требования ФГОС (ОК-8,9,16, ПК-5, 11), Критерий 5 АИОР (5.2.14), согласованный с требованиями международных стандартов *EUR-ACE* и *FEANI.* Требования работодателей: Контек, ОАО «Газпром переработка», ООО Нижневартовскэнергонефть».

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники
Направление подготовки: Прикладная математика и информатика
Отделение информационных технологий

УТВЕРЖДАЮ:
Зав. кафедрой

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

бакалаврской работы

Студенту:

Группа	**ФИО**
8Б51	Кружков Денис Сергеевич

Тема работы:

Решение задачи квадратичного программирования на основе операторов-проекторов	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования; документы конференции и отчеты НИР; программное обеспечение).*	Матрицы и векторы коэффициентов квадратичной и линейной части целевого функционала и ограничений.
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных*	1) Аналитический разбор литературы по теме “квадратичное программирование”. 2) Постановка задачи. 3) Изучение вопросов и тем, необходимых для понимания и использования алгоритма. 4) Изучение и обоснование алгоритма для решения задач квадратичного программирования

разделов, подлежащих разработке; заключение по работе).	5) Использование и доработка алгоритма решения задач квадратичного программирования, составление программы. 6) Тестирование работоспособности алгоритма и программы. 7) Применение программы для тестового примера. 8) Заключение к работе
Перечень графического материала *(с точным указанием обязательных чертежей)*	

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
1. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Подопригора Игнат Валерьевич
2.Социальная ответственность	Мезенцева Ирина Леонидовна

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Вылегжанин Олег Николаевич	к.х.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8Б51	Кружков Денис Сергеевич		

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт кибернетики
Направление подготовки Прикладная математика и информатика
Уровень образования бакалавр
Кафедра программной инженерии
Период выполнения осенний / весенний семестр 2018/2019 учебного года

Форма представления работы:

бакалаврская работа

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
08.02.2019	Постановка целей и задач, получение исходных данных	
16.02.2019	Подбор и изучение материалов по тематике	
27.03.2019	Разработка календарного плана	
01.03.2019	Обсуждение литературы	
11.03.2019	Выбор метода для реализации	
27.03.2019	Изучение и доработка алгоритма решения задачи	
10.04.2019	Реализация алгоритмов в виде программы	
30.04.2019	Проведение расчетов	
14.05.2019	Оформление расчётно-пояснительной записки	
21.05.2019	Оформление графического материала	
28.05.2019	Подведение итогов	

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Вылегжанин Олег Николаевич	к.х.н		

СОГЛАСОВАНО:

	ФИО	Ученая степень, звание	Подпись	Дата

РЕФЕРАТ

Пояснительная записка выполнена на 96 листах машинописного текста, содержит 10 иллюстраций, 12 таблиц, 24 источника, 11 приложений.

Ключевые слова: квадратичное программирование, оператор-проектор, ограничения-равенства, ограничения-неравенства, область допустимых значений,

Работа посвящена реализации алгоритма поиска условного оптимума задачи квадратичного программирования с системой смешанных линейных ограничений.

Объектом исследования является целевой функционал, состоящий из квадратичной и линейной части, а также система смешанных ограничений, образующая пространство решений задачи.

Целью работы является изучение и доработка алгоритма поиска условного экстремума целевого функционала в области допустимых значений, образованной наложенными линейными ограничениями.

В процессе работы был изучен алгоритм учёта ограничений-равенств, а также алгоритм поиска оптимального решения с учётом оператора-проектора. Алгоритм и программа были проверены на тестовом примере, подтверждающей их корректность.

Область применения: математика, системный анализ, техника, экономика, медицина и статистика.

Экономическая эффективность работы заключается в сокращении времени, необходимого для решения задачи подобного типа, путем использования программы.

ОПРЕДЕЛЕНИЯ

В данной работе используются следующие термины с соответствующими определениями:

целевой функционал – функция, подлежащая оптимизации в рамках поставленной задачи;

оператор-проектор – матрица, построенная для заданного линейного пространства, результатом умножения произвольного вектора на которую является ортогональная пространству компонента этого вектора;

область допустимых значений – пространство решений задачи, образованное наложенными ограничениями;

гиперплоскость – пространство, имеющее размерность на единицу меньшую, чем исходное пространство;

псевдообратная матрица – обобщённая обратная матрица;

градиент функции – вектор, координатами которого являются частные производные функции;

рекуррентность – вычисляемость на основе предыдущих значений.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

К задачам на поиск оптимума сводятся многие из проблем математики, системного анализа, техники, экономики, медицины и статистики [1]. Любая задача оптимизации подразумевает собой наличие определённого набора независимых переменных, а также условий, которые характеризуют их приемлемые значения. Такие условия называют ограничениями задачи. Ещё одной обязательной частью такой задачи является целевой функционал, зависящий от переменных. Решением задачи оптимизации является приемлемый набор значений независимых переменных, которому удовлетворяет оптимальное значение упомянутой выше целевой функции. Под оптимальностью же обычно понимают её максимальность или минимальность.

Рассматриваемое в данной работе квадратичное программирование – это особый тип задачи математической оптимизации, обусловленный наличием квадратичного функционала, множество возможных решений для которого задано линейными ограничениями.

Существует большое количество методов для нахождения оптимального значения целевого функционала, описанных в научной литературе, однако единого способа, отвечающего всем требованиям эффективности, разработано не было. Поэтому в ходе выполнения работы, алгоритм решения задачи квадратичного программирования может быть разделён на следующие ключевые этапы, где для решения некоторых из них существуют эффективные алгоритмы:

– исследование и реализация алгоритма учёта ограничений равенств;

– исследование и реализация алгоритма поиска базисного решения

– исследование и разработка с последующей реализацией алгоритма поиска оптимума на основе метода «Операторов-проекторов»;

– исследование отлаженного алгоритма на результативность и безошибочность выполнения с использованием тестовых примеров.

Объектом исследования в рамках выполнения данной работы является алгоритм поиска оптимального значения целевого функционала. Предмет

исследования – это свойства, характеризующие результативность и безошибочность выполнения исследуемого алгоритма.

ОБЗОР ЛИТЕРАТУРЫ

Задача решения квадратичного программирования имеет множество приложений в различных проблемных областях, несмотря на это, она имеет слабое освещение в литературе.

Основополагающей книгой по данной тематике является монография [1]. Излагаемая в [1] теория охватывает все основные аспекты квадратичного программирования. В книге [1] собрано большинство методов, созданных к моменту ее написания, для каждого из которых указаны условия применения.

В статьях [2], [3] приведено и обосновано использование оператора-проектора для задачи линейного программирования, а также процедура учета системы ограничений равенств для понижения размерности системы линейных неравенств.

В [4] дано краткое и простое изложение теории оптимизации, а также некоторые ее приложения.

В [5] дано полное изложение основ теории линейного программирования, которое является краеугольным камнем теории квадратичного программирования.

В статье [10] описана часть методов решения систем линейных неравенств, а именно получение одной из вершин многогранника решений посредством итерационной процедуры с применением оператора-проектора.

1 МАТЕМАТИЧЕСКАЯ ПОСТАНОВКА ЗАДАЧИ КВАДРАТИЧНОГО ПРОГРАММИРОВАНИЯ

Пусть $F(x)$ – квадратичная выпуклая функция, которая в дальнейшем обозначим через $I(x)$:

$$I(x) = x^T Q x + p^T x. \quad (1.1)$$

Здесь Q – симметричная положительно полуопределённая матрица. Пусть, далее,

$$f_i(x) = {a_i}^T x - b_i. \quad (1.2)$$

Сформулируем основную задачу квадратичного программирования с линейными условиями как задачу минимизации $I(x)$ при ограничениях

$${a_i}^T x - b_i = 0,\ i = 1,2,\dots,m, \quad (1.3)$$

$$x \geq 0. \quad (1.4)$$

Если в квадратичных задачах, встречающихся на практике, целевой функционал не минимизируется, а максимизируется или, если в некоторых ограничениях вместо знака «≤» стоит знак «≥», то такие задачи всегда можно привести к вышеупомянутой форме. Для этого достаточно умножить на -1 целевой функционал и соответствующие неравенства [1].

Если m векторов строк ${a_i}^T$ собрать в матрицу A с размерами $m \times n$ и величины b_i – в вектор b, то систему ограничений можно записать в виде

$$A_1 x - b_1 = 0. \quad (1.5)$$

Эти ограничения являются ограничениями-равенствами. Аналогично, ограничения-неравенства имеют вид

$$A_2 x - b_2 \leq 0. \quad (1.6)$$

И, тогда, квадратичную задачу можно представить в матричной записи:

$$\min\{I(x)|A_1x = b_1, A_2x \leq b_2, x \geq 0\}. \quad (1.7)$$

Постановка двойственной задачи в матричной форме выглядит как:

$$\max\{I(y)|A_1y = b_1, A_2y \geq b_2, y \geq 0\} \quad (1.8)$$

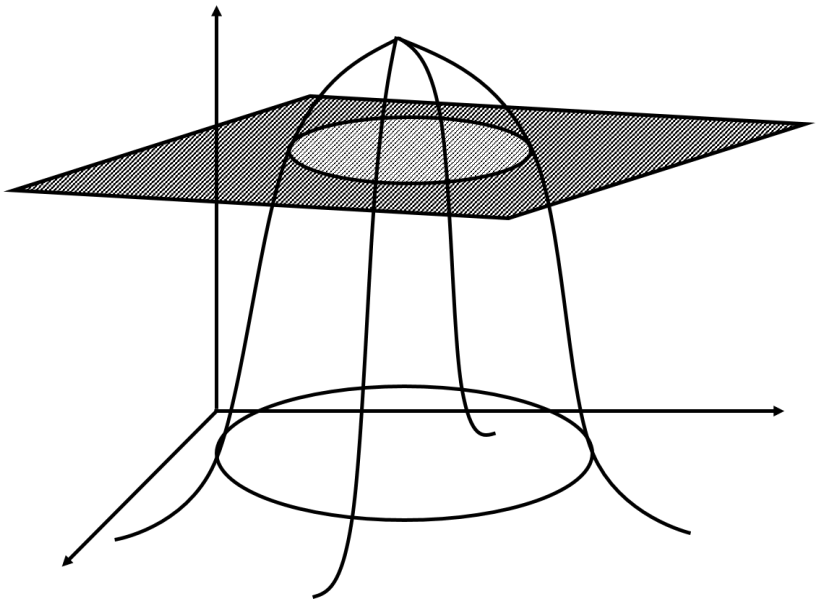

Рисунок 1 – Геометрическая интерпретация задачи

Элементы A, b и p могут быть заданы произвольно.

Одним из наиболее популярных методов решения задач на поиск условного экстремума с ограничениями в виде равенств заключается в замене исходного целевого функционала на функцию Лагранжа, которая формируется на основе уравнений связи [1].

Функция Лагранжа для данной задачи имеет вид

$$\Phi(x, \lambda) = I(x) + \sum_{i=1}^{m} \lambda_i({a_i}^T x - b_i) = x^T Q x + p^T x + \lambda^T(Ax - b). \quad (1.9)$$

Обозначим

$$\frac{\partial \Phi}{\partial x} = v, \quad -\frac{\partial \Phi}{\partial \lambda} = y, \quad (1.10)$$

тогда

$$v = 2Qx + p + A^T\lambda = \frac{\partial \Phi}{\partial x}, \quad (1.11)$$

$$y = -Ax + b = -\frac{\partial \Phi}{\partial \lambda}. \quad (1.12)$$

Таким образом решение (1.9) совпадает с условным экстремумом целевого функционала с учётом наложенных на него ограничений-равенств, т.е. задача

условной оптимизации сводится к поиску решения функции Лагранжа без ограничений.

Определение точки оптимума занимает центральное место в теории квадратичного программирования. Для проверки точки на экстремальность на основе обобщения классического метода множителей Лагранжа была разработана теорема Куна-Таккера, из которой, в последствии, были сформированы условия, определяющие экстремум при наличии ограничивающих условий.

Условия Куна-Таккера для задачи квадратичного программирования определяются теоремой Куна-Таккера, описанной в [1] и имеют вид:

$$Ax + y = b, \tag{1.13}$$

$$2Qx - v + A^T\lambda = -p, \tag{1.14}$$

$$x^T v + y^T \lambda = 0, \tag{1.15}$$

$$x \geq 0, v \geq 0, y \geq 0, \lambda \geq 0. \tag{1.16}$$

Выполнение условий (1.11) – (1.14) необходимо и достаточно для существования решения квадратичной задачи. Другими словами, если существует такой x^*, для которого выполняются условия Куна-Таккера, то x^* – искомое решение.

2 АНАЛИЗ ГЕОМЕТРИИ ЗАДАЧИ КВАДРАТИЧНОГО ПРОГРАММИРОВАНИЯ

2.1 Неограниченный экстремум задачи квадратичного программирования

Безусловную задачу квадратичного программирования можно сформулировать как

$$\min\{I(x)\} = \min\{x^T Q x + p^T x\}, \quad (2.1.1)$$

где $I(x)$ – выпуклая дифференцируемая функция.

Тогда для того, чтобы решением задачи (2.1.1) являлся элемент $x^* \in R^n$, необходимо и достаточно чтобы градиент минимизируемой функции $I(x)$ в точке экстремума был равен нулю:

$$\nabla I(x^*) = \frac{\partial I(x^*)}{\partial x} = 2Qx^* + p = 0. \quad (2.1.2)$$

Покажем, что (2.1.2) верно, если $I(x)$ выпукла:

$I(x) - I(x^*) \geq \left(\frac{\partial I(x^*)}{\partial x}, x - x^*\right) = 0$, для $\forall x \in R^n$,

т.е. $I(x) \geq I(x^*)$ и x^* – решение (2.1.1).

2.2 Геометрический смысл решения задачи квадратичного программирования

В задаче квадратичного программирования целевая функция в пространстве представляет собой некоторую поверхность, определённую на множестве заданных переменных. Такое множество определяется ограничениями-неравенствами.

Каждое ограничение представляет собой $(n-1)$-мерную гиперплоскость, которая делит исходное пространство на два подпространства, одно из которых должно удовлетворять данному условию. Система ограничений, в таком случае, определяет выпуклая область допустимых значений – общую часть n-мерного пространства, удовлетворяющую всем ограничениям.

Все имеющиеся ограничения образуют выпуклое множество размерности n. В случае $n = 2$, данное множество представляет собой выпуклый многогранник ограничений (рис. 2.2.1).

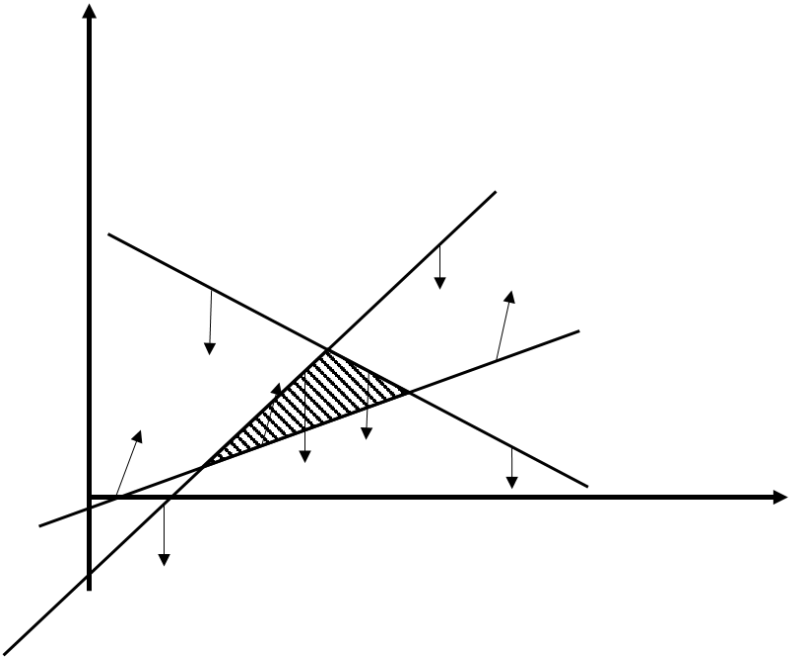

Рисунок 2.2.1 – Многогранник ограничений

В случае, когда решение задачи квадратичного программирования единственно, рассматриваемый многоугольник ограничений является точкой, прямой или отрезком, в зависимости от размерности решаемой задачи.

Рассмотрим геометрическую интерпретацию квадратичной задачи. Пусть матрица Q положительно определена. В этом случае $I(x)$ строго выпукла, линии равного уровня $I(x) = const$ образуют эллипсы, общий центр которых соответствует безусловному минимуму $I(x)$. Нахождение решения сводится к определению той точки многогранника ограничений, которая лежит на линии уровня с наименьшим значением $I(x)$ [1].

При рассмотрении двумерного случая возникает одна из трёх возможных ситуаций, когда

а) искомое решение находится на пересечении двух рёбер многоугольника ограничений (рис. 2.2.2, а);

б) условный экстремум находится на одном из рёбер многоугольника ограничений (рис. 2.2.2, б);

в) решение задачи находится внутри многоугольника допустимых значений (рис. 2.2.2, в).

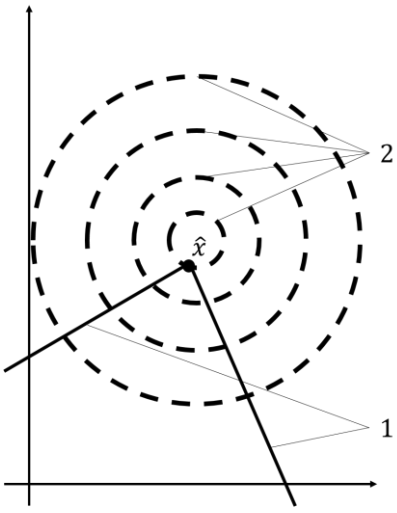

Рисунок 2.2.2, а

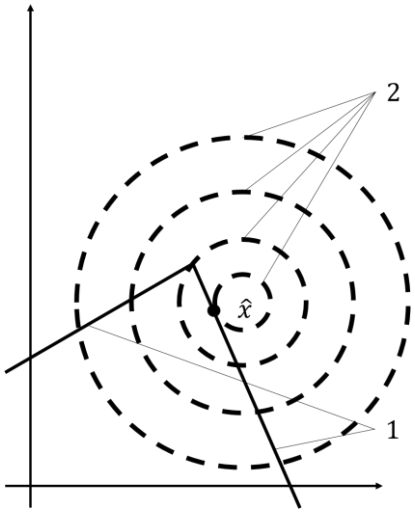

Рисунок 2.2.2, б

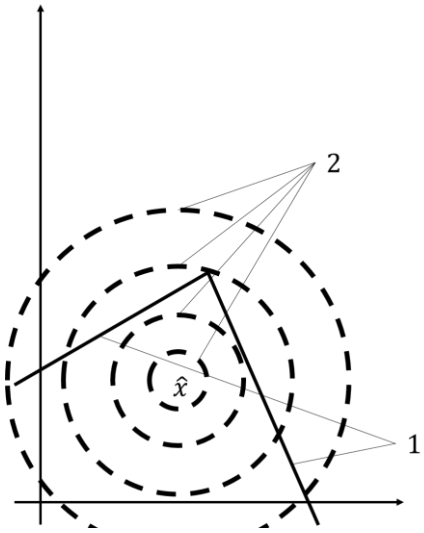

Рисунок 2.2.2, в

1 – рёбра многоугольника ограничений,

2 – линии уровня целевой функции $I(x)$,

$\hat{x}$ – искомое решение задачи КП.

2.3 Роль ограничений-равенств и ограничений-неравенств

В случае задачи квадратичного программирования, ограничения, наложенные на целевой функционал, часто имеют смешанный характер и определяются в виде системы, состоящей из уравнений (2.3.1) и неравенств (2.3.2).

$$A_{1_1}x_1 + \cdots + A_{1_n}x_n = b_1 \qquad (2.3.1)$$

$$A_{2_1}x_1 + \cdots + A_{2_n}x_n \leq b_2 \qquad (2.3.2)$$

$$x_1 \geq 0, \ldots, x_n \geq 0 \qquad (2.3.3)$$

Каждое неравенство такой системы геометрически определяет полупространство с граничной прямой $a_{2i_1}x_1 + \cdots + a_{2i_n}x_n = b_{2i}, i = \overline{1,m}$. Условия неотрицательности (3.3.3) определяют полуплоскости в соответствии с граничными прямыми $x_1 = 0, \dots, x_n = 0$.

Пересечение выпуклых плоскостей образует их общую часть, которая сама является выпуклым множеством и представляет собой совокупность точек, координаты каждой из которых составляют решение смешанной системы (2.3.1)-(2.3.3). Совокупность полученных точек называют областью допустимых значений.

В случае системы, где $n = 3$, каждое ограничение-неравенство делит исходное трёхмерное пространство на два полупространства, граничная плоскость каждого из которых $a_{2i_1}x_1 + a_{2i_2}x_2 + a_{2i_3}x_3 = b_{2i}$, а условия неотрицательности – полупространства с граничными плоскостями $x_1 = 0, x_2 = 0, x_3 = 0$. В пространстве, размерности $n = 3$, пересекаясь, такие множества образуют один из трёх возможных видов области допустимых значений:

а) многогранник;

б) конус;

в) пересечение многогранника и конуса.

Таким образом, геометрический смысл решения задачи квадратичного программирования состоит в отыскании такой точки, принадлежащей области допустимых значений, сформированной из ограничений-неравенств, координаты которой доставляют минимальное значение целевому функционалу.

Рассмотрим более детально геометрическую интерпретацию ограничений-равенств в задаче квадратичного программирования. Каждое из таких ограничений представляет собой пространство размерности $n - 1$, а пересечение двух таких плоскостей уменьшает размерность пространства на единицу до $n - 2$. Следовательно, роль ограничений-равенств состоит в уменьшении размерности пространства, которому принадлежит определяемый вектор.

3 АЛГОРИТМ УЧЁТА ОГРАНИЧЕНИЙ-РАВЕНСТВ

В случае задачи квадратичного программирования предполагается наличие ограничений-равенств вида (1.5).

Набор ограничений-равенств позволяет снизить размерность пространства решений задачи на число, равное рангу матрицы A_1 из (1.5).

Для учёта ограничений, выполняющихся как строгое равенство, необходимо построить такой оператор, в результате воздействия на матрицу ограничений-равенств которого полученная матрица имела вид (3.1).

$$PA = |DA'|, \tag{3.1}$$

где D – диагональная матрица, имеющая размер (k, k), а A' – некоторая матрица размером $(k \cdot (n-k))$. Целью такого преобразования является выражение первых k неизвестных столбцов как линейных функций от остальных.

Теорема 3.1 Если A_1 – матрица ограничений-равенств, ранг матрицы $rankA = k$, и первые k столбцов A_1 линейно независимы, то

$$\hat{x} = \hat{A}^{+}(b - \bar{A} \cdot \bar{x}), \tag{3.2}$$

где $\hat{x}$ – вектор первых k элементов, $\bar{x}$ – вектор остальных элементов x, а $\hat{A}^{+}$ – матрица, псевдообратная к первым k столбцам A_1. Данная теорема приведена и доказана в [2].

Учёт ограничений-равенств осуществляется с использованием рекуррентного псевдообращения, описанного в [2]. Схема алгоритма представлена на рисунке 3.1.

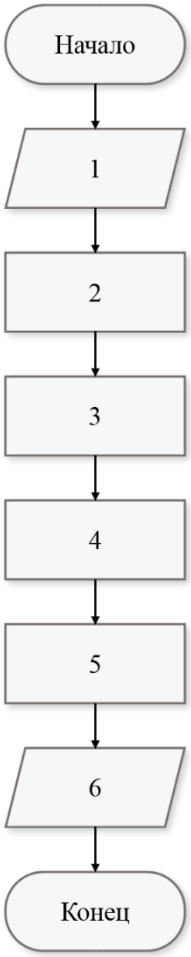

Рисунок 3.1 – Блок-схема общего алгоритма учёта ограничений-равенств

1) Ввод начальных данных: матрица коэффициентов квадратичной части функционала Q, вектор коэффициентов линейной части функционала p, матрица ограничений-равенств A_1, матрица ограничений-неравенств A_2, вектор свободных элементов ограничений-равенств b_1, допустимая погрешность tol.

2) Расчёт псевдообратной матрицы A_1^+ для матрицы ограничений-равенств A_1.

2.1) Поиск столбца с максимальной нормой матрицы A_1 с помощью формулы

$$\|A_{1_i}\| = \sqrt{{A_{1_i}}^T \times A_{1_i}} \tag{3.3}$$

2.2) Запись найденного столбца в новую матрицу An и удаление его из матрицы ограничений A_1. Элементы с соответствующим номером удаляются из A_2 и p.

2.3) Расчёт псевдообратной матрицы An^+ по формуле

$$An^+ = \frac{An^T}{An^T \times An} \tag{3.4}$$

3) Процесс рекуррентного псевдообращения.

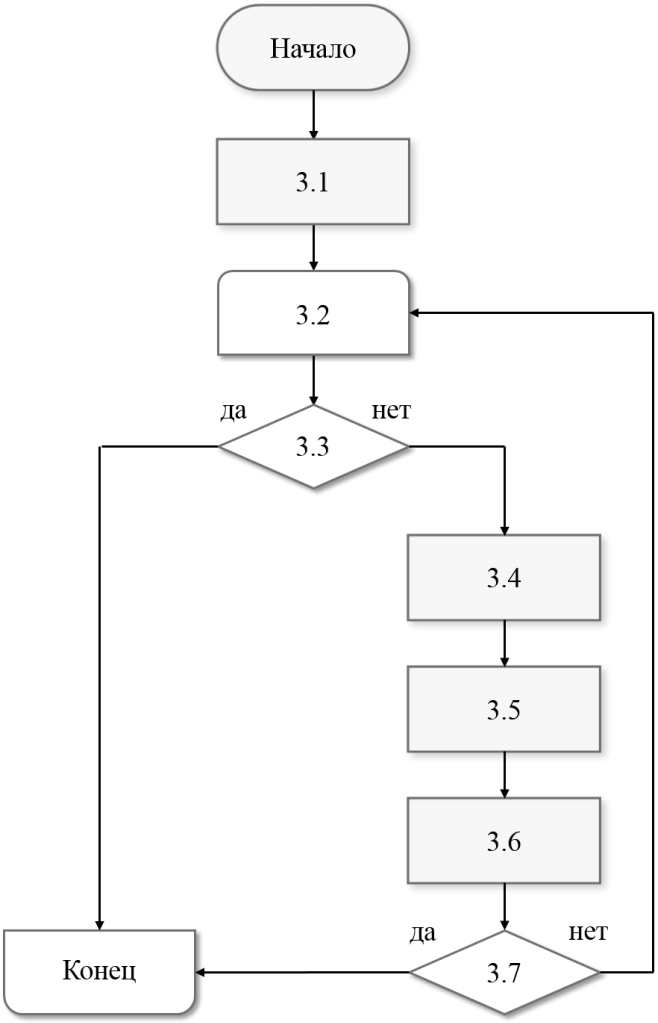

Рисунок 3.2 – Блок-схема рекуррентного псевдообращения

3.1) Поиск вспомогательной матрицы C по формуле

$$C = A - An \times An^{+} \times A \quad (3.5)$$

3.2) Поиск столбца с максимальной нормой матрицы C.

3.3) Проверка условия: превышает ли норма найденного столбца заданную наперёд точность tol.

3.4) Удаление из матрицы A_1, A_2 и p и запись из A_1 в An.

3.5) Расчёт псевдообратной матрицы к $C - C^{+}$.

3.6) Пересчёт An^{+} на основе C^{+}.

3.7) Проверка условия: не пуста ли матрица A_1

4) Пересчёт коэффициентов целевой функции.

4.1) Разбиение матрицы Q по следующему правилу:

$$Q = \begin{bmatrix} Q_0 & \hat{Q} \\ \bar{Q} & \tilde{Q} \end{bmatrix} \quad (3.6)$$

4.2) Пересчёт матрицы Q по формуле:

$$Q = W^T Q_0 W - \bar{Q} W + W^T \hat{Q} + \tilde{Q}, \quad (3.7)$$

где $W = An^{+} \times A_1$.

4.3) Пересчёт вектора p по формуле:

$$p = p^T - pn^T An^{+} A_1 + {b_1}^T * (\tilde{Q}^T + \hat{Q} - (Q_0 + {Q_0}^T) W). \quad (3.8)$$

5) Пересчёт вектора свободных элементов b_1 по формуле:

$$b_1 = An^{+} \times b_1 \quad (3.9)$$

Вывод преобразованных матриц Q, An, A_2 и векторов p и b_1.

В результате выполнения алгоритма учёта ограничений-равенств обеспечивается выбор столбцов, входящих в матрицу ограничений A_1, образующих базисный набор для всех столбцов такой матрицы, и вычисление матрицы, псевдообратной к матрице, образованной этими столбцами[2].

4 ПОСТАНОВКА ЗАДАЧИ КВАДРАТИЧНОГО ПРОГРАММИРОВАНИЯ ПРИ НАЛИЧИИ ТОЛЬКО ОГРАНИЧЕНИЙ-НЕРАВЕНСТВ

После выполнения алгоритма по учёту ограничений, заданных уравнениями, исходный функционал задачи $I(x)$ изменяется следующим образом в соответствии с (3.4)-(3.6):

$$I'(x) = x^T Q' x + {p'}^T x, \tag{4.1}$$

где Q' – преобразованная матрица коэффициентов квадратичной части целевого функционала, а p' – преобразованный вектор коэффициентов линейной части.

Теорема 4.1 Если $<p, x>$ – линейная форма, то при наличии ограничений-равенств вида (1.5) она преобразуется к форме:

$$p^T \cdot x = \hat{p}^T \hat{A}^+ b + \left(\bar{p}^T - \hat{p}^T \hat{A}^+ \bar{A}\right) \cdot \bar{x}, \tag{4.2}$$

где $\hat{p}$ и $\bar{p}$ – компоненты вектора коэффициентов, соответствующие компонентам $\hat{x}$ и $\bar{x}$ вектора x.

Теорема 4.2 Если $x^T Q x$ – квадратичная форма, то при наличии ограничений-равенств вида (1.5) она преобразуется к форме:

$$x^T Q x = \bar{x}^T \left(W^T Q_0 W - \hat{Q} W + W^T \bar{Q} + \tilde{Q}\right)\bar{x} +$$
$$+\bar{b}^T \left(\hat{Q}^T + \bar{Q} - \left(Q_0 + {Q_0}^T\right) W\right)\bar{x} + \bar{b}^T Q_0 \bar{b} \tag{4.3}$$

Здесь $W = \hat{A}^+ A$, $\bar{b} = \hat{A}^+ b$, а Q_0, $\bar{Q}$, $\hat{Q}$ и $\tilde{Q}$ – компоненты матрицы квадратичной формы размеров $(k \cdot k)$, $(k \cdot (n-k))$, $((n-k) \cdot k)$ и $((n-k) \cdot (n-k))$ соответственно как это показано в (4.4).

$$Q = \begin{vmatrix} Q_0 & \bar{Q} \\ \hat{Q} & \tilde{Q} \end{vmatrix} \tag{4.4}$$

Также наряду с целевым функционалом задачи изменяется и матрица ограничений неравенств таким образом, что размерность пространства, в котором определена область допустимых решений, сокращается до $n-k$, где k – ранг матрицы ограничений-равенства A_1. Доказательство данной теоремы приведено в [2].

5 ПОИСК ОПОРНОГО РЕШЕНИЯ

Для перехода к этапу поиска условного оптимума целевой функции необходимо переместить рабочую точку из случайных координат в вершину фигуры области допустимых значений. Такой переход осуществляется в два этапа.

При задании начальной точки x_0, из которой будет осуществляться движение по направлению к условному экстремуму, наиболее распространённым случаем является ситуация, когда x_0 лежит за пределами области допустимых значений. Если ближайшая к начальной точке поверхность области допустимых значений имеет направляющим вектором a, то проекцию начальной точки – x_1на такое полупространство можно найти, используя теорему 5.1.

Теорема 5.1 Если x_0 – начальная точка, а ближайшая к ней плоскость многогранника M имеет направляющим вектором a_i, то x_1 – проекция начальной точки на эту плоскость есть:

$$x_1 = x_0 + \frac{b_i - a_i^T x_0}{a_i^T a_i} * a_i. \quad (5.1)$$

Таким образом, переместившись на область допустимых значений в точку ближайшую к x_0, необходимо для всех компонент i, для которых

а) $y_i = b_i - a_i x_k \leq 0;$ (5.2)

б) $\frac{b_i - a_i^T x_k}{a_i^T a_i}$ – минимален (5.3)

найти проекцию полученной точки на пересечение полупространства, заданного столбцом a_i, с пересечением полупространств, заданных столбцами, составляющими матрицу ограничений-неравенств A_2 с помощью теоремы(5.2) [3].

Теорема 5.2 Если точка x_k принадлежит пересечению k плоскостей, заданных направляющими векторами $a_j|j = 1, \dots, k$, то ее проекция на пересечение плоскости, заданной строкой a_i, с пересечением плоскостей, заданных строками $a_j|j = 1, \dots, k$, составляющими матрицу A_2, есть:

$$x_{k+1} = x_k + \frac{b_i - a_i^T x_k}{a_i^T R a_i} * R * a_i, \qquad (5.4)$$

где $R = (I - A_2 \times A_2^+)$ – оператор-проектор на пространство, перпендикулярное пространству, натянутому на столбцы матрицы ограничений-неравенств [3].

В результате выполнения вышеупомянутых двух этапов получим точку, для которой $\forall y_i \leq 0 \,|i = 0, \dots, n$, т. е. являющуюся вершиной многогранника M.

6 ИТЕРАЦИОННЫЙ ПРОЦЕСС ПОИСКА РЕШЕНИЯ МЕТОДОМ ОПЕРАТОРОВ-ПРОЕКТОРОВ

Переход от базисного решения к оптимальному начинается с проверки текущего положения рабочей точки на принадлежность к условному экстремуму при помощи проверки условий Куна-Таккера (1.12)-(1.15). Проверка такого рода сводится к следующему рекуррентному правилу: пусть рабочая точка x_k принадлежит ровно $q(0 \le q \le n-k)$ ограничивающим гиперплоскостям, заданным ограничениями-неравенствами, тогда:

Вычисляется $\nabla I(x_k)$, $U = {A_2}^{+^T} \cdot \nabla I(x_k)$ и $R_q \cdot \nabla I(x_k)$, где R_q – оператор проектор на q гиперплоскостей. Если $R_q \cdot \nabla I(x_k) = 0$, и $U \ge 0$, то такое x_k – есть искомое решение.

Если условия не выполнены, поиск решения задачи квадратичного программирования осуществляется путём перемещения по поверхности области допустимых значений в направлении градиента целевого функционала (2.1.2).

$$x_1 = x_0 + \alpha * \nabla I(x) = x_0 + \alpha * (2Qx + p) \quad (6.1)$$

Представим искомое решение в области допустимых значений задачи как

$$A_2 * x^* = x^T Q x + p^T x, \quad (6.2)$$

тогда, при условии (2.1.2), решение можно преобразовать к следующему выражению:

$$x^* = -Q^+ \times p. \quad (6.3)$$

Используя оператор-проектор R, распишем матрицу Q и вектор p:

$$Q = R^T \times Q \times R, \quad (6.4)$$

$$p = R \times \nabla I(x). \quad (6.5)$$

Подставим полученные выражения (6.4)-(6.5) в (6.3):

$$x^* = (R^T \times Q \times R)^+ \times \left(R \times \nabla I(x)\right) = R^T \times Q \times \nabla I(x), \quad (6.6)$$

где x^* – искомое решение[1].

Таким образом, посредством формулы (6.5) возможно осуществление перемещения рабочей точки по области допустимых значений по направлению к улучшению (минимизации) целевого функционала.

7 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА

7.1 Выбор программной среды

Для реализации разработанных алгоритмов могут быть использованы такие типы программного обеспечения, как специализированные математические пакеты (MATLAB, MathCad) или различные объектно-ориентрованные языки программирования (C++, Pascal, Delphi).

Преимуществами математических пакетов является запись сложных математических выражений в исходном виде, оптимизированные алгоритмы численных и аналитических расчётов, а также возможность создания встроенными средствами высококачественных таблиц, графиков и текстовых документов. Однако, большая часть программного обеспечения такого типа ограничена базовым функционалом или имеет узкую специализацию в программировании сложных задач.

Для решения таких задач, как создание готового продукта для проведения вычислений в области математического программирования, наиболее подходящим вариантом являются объектно-ориентированные языки программирования. Такие языки содержат необходимые средства для создания программ любого назначения: от низкоуровневых до сложнейших программных комплексов.

7.2 Преимущества выбранной среды разработки

В качестве средства реализации алгоритма решения задачи квадратичного программирования была выбрана интегрированная среда разработки Microsoft Visual Studio в совокупности с языком C++. Данная среда имеет ряд преимуществ по отношению к конкурирующему программному обеспечению, таких как:

- встроенная функция управления исходным кодом;
- возможность рефакторизации кода;

– мощная модель расширяемости за счёт подключения различных библиотек (в том числе и математических);

– более интуитивные инструменты разработки и отладки реализовываемого продукта;

– поддержка множества языков программирования (в том числе и C++).

Использование выбранного программного обеспечения позволяет минимизировать ограничения, наложенные средой разработки на реализацию алгоритмов решения задачи квадратичного программирования.

7.3 Ключевые этапы реализации алгоритма

Процесс поиска оптимального значения минимизируемого функционала представляет собой сложный алгоритм, состоящий из множества этапов. Однако в структуре решения задачи квадратичного программирования часто встречаются повторяющиеся базовые действия с элементами линейной алгебры, такие как: переменожение матриц, поиск обратной матрицы, поиск псевдообратной матрицы, разбиение матрицы на две на основе определённого критерия и т.д. В Visual Studio C++ не предусмотрено встроенного функционала для работы с такими операциями. Для уменьшения объёмов программной реализации поиска решения базовые действия над элементами линейной алгебры были объединены и вынесены за основное тело программы как внешние функции.

7.3.1 Перемножение матриц

Одним из основных действий с матрицами является их перемножение. Реализованный алгоритм перемножения матриц (приложение А) используется на протяжении выполнения всей программной реализации решения задачи квадратичного программирования: как в основном теле программы, так и во внешних функциях.

Данный алгоритм является наипростейшей матричной операцией и реализован по привычной схеме перемножения матриц, рассматриваемой в курсе «Линейной алгебры».

7.3.2 Разбиение матрицы

Разбиение матрицы (приложение Б) предполагает два основных действия: копирование определённых столбцов в новую матрицу и удаление этих столбцов из исходной.

Реализация удаления столбцов из исходной матрицы представляет собой смещение столбцов «влево» по индексу и последующее зануление «лишнего» столбца, который является дубликатом последнего смещённого.

7.3.3 Поиск обратной матрицы

Процесс поиска обратной матрицы (приложение В) представляет собой базовую операцию курса «Линейная алгебра». Обратная матрица ищется по формуле:

$$A^{-1} = \begin{pmatrix} \frac{M_{11}}{detA} & \cdots & \frac{M_{n1}}{detA} \\ \vdots & \ddots & \vdots \\ \frac{M_{1n}}{detA} & \cdots & \frac{M_{nn}}{detA} \end{pmatrix}, \quad (7.3.3.1)$$

где $detA$ – определитель матрицы A, M_{nn} – минор матрицы A.

7.3.4 Поиск псевдообратной матрицы

Псевдообратная матрица – есть обобщённая обратная матрица к исходной. Разработанная функция по отысканию псевдообратной матрицы реализована на основе сингулярного разложения матриц.

Исходную матрицу A можно разложить следующим образом:

$$A = U\lambda V^T, \quad (7.3.4.1)$$

Где U и V – унитарные матрицы, а λ – диагональная матрицы сингулярных чисел.

Тогда псевдообратную матрицу к A можно представить следующим образом:

$$A^+ = U\lambda^{-1}V^T. \quad (7.3.4.2)$$

Функция использует библиотеку svd.h (а также nr3.h), которая позволяет получить матрицы U, λ и V сингулярного разложения.

7.3.5 Поиск столбца с максимальной нормой

Норма столбца находится по формуле:

$$\|a\| = \sqrt{a^T \times a}. \quad (7.3.4.1)$$

Функция работает по принципу поиска столбца с максимальным значением нормы заданной матрицы и запоминания номера такого столбца. Поиск максимального осуществляется посредством сравнения нормы проверяемого столбца с текущим значением максимальной нормы.

Листинг программной реализации данного алгоритма во внешней функции представлен в приложении Г.

7.3.6 Вывод матрицы на экран

Данная функция представляет собой обобщённый алгоритм поэлементного вывода на экран значений матрицы с переносом строк там, где это необходимо для получения внешнего вида матрицы, сопоставимого с представлением матриц в курсе «Линейной алгебры».

Функция вынесена за тело программы для сокращения объёма кода реализации алгоритма и автоматизации вывода данных на экран (приложение Д).

8 РЕШЕНИЕ ТЕСТОВОГО ПРИМЕРА

Для проверки корректности работы реализованного алгоритма необходимо произвести расчёты на тестовом примере.

Матрицы квадратичной части целевого функционала (Q), линейной части целевого функционала (P), ограничений-равенств (A_1) и ограничений-неравенств (A_2), а также соответствующие им векторы свободных элементов b_1 и b_2 имеют вид:

$$Q = \begin{bmatrix} 5 & 3 & 2 & 0 & 1 \\ 3 & 3 & -4 & 2 & -3 \\ 2 & -4 & 4 & -2 & 4 \\ 0 & 2 & -2 & 6 & -3 \\ 1 & -3 & 4 & -3 & 1 \end{bmatrix}, p = \begin{bmatrix} 1 \\ -2 \\ 1 \\ -3.424 \\ 0.5 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} 1.2 & -3 & 1 & 0 & -1 \\ 1.4 & -1.5 & 1 & -1 & -1 \\ -0.7 & 0.5 & 1 & -1 & -1 \\ 0.26 & -0.45 & 0.2 & -0.1 & -0.2 \end{bmatrix}, b_1 = \begin{bmatrix} -0.5 \\ 0.5 \\ 4.5 \\ 0 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0.4 & -2 & 0.1 & 0.501 & 0.35 \\ -1 & 0.1 & -2 & 9.994 & -3.575 \\ -0.5 & 0.1 & 0.5 & -7.481 & 5.425 \\ 4.5 & -3 & -10.1 & -16.551 & -30.05 \\ 4.5 & -3 & 0 & -2.398 & -5 \\ -1 & 2 & 0.5 & -1.2 & -0.35 \\ 2.4 & -1 & -3.5 & 4 & -5.2 \\ -1 & 1 & -3 & 4.687 & -1.25 \end{bmatrix}, b_2 = \begin{bmatrix} 2.225 \\ 11.037 \\ 6.537 \\ 110.225 \\ -12 \\ 1.125 \\ 0.2 \\ -2.625 \end{bmatrix}.$$

После учёта ограничений (глава 3) равенств матрицы коэффициентов целевого функционала, а также матрицы ограничений будут иметь вид:

$$Q' = \begin{bmatrix} 1.0585 & 0.232293 & 2.32653 \\ 0.232293 & 0.0807329 & -0.317051 \\ 2.32653 & -0.317051 & 8.7997 \end{bmatrix}, p' = \begin{bmatrix} -3.93301 \\ -16.3495 \\ 27.8744 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} 1.2 & 1 & -1 \\ 1.4 & 1 & -1 \\ -0.7 & 1 & -1 \\ 0.26 & 0.2 & -0.2 \end{bmatrix}, b_1 = \begin{bmatrix} -0.5 \\ 0.5 \\ 4.5 \\ 0 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0.4 & 0.1 & 0.35 \\ -1 & -2 & -3.575 \\ -0.5 & 0.5 & 5.425 \\ 4.5 & -10.1 & -30.05 \\ 4.5 & 0 & -5 \\ -1 & 0.5 & -0.35 \\ 2.4 & -3.5 & -5.2 \\ -1 & -3 & -1.25 \end{bmatrix}, b_2 = \begin{bmatrix} 2.225 \\ 11.037 \\ 6.537 \\ 110.225 \\ -12 \\ 1.125 \\ 0.2 \\ -2.625 \end{bmatrix}.$$

Таким образом, после учёта ограничений-равенств, исходная задача преобразуется из задачи пространства $n = 5$ к задаче в трёхмерном пространстве ($n = 3$).

После сокращения размерности пространства становится возможным реализовать графическую интерпретацию ограничений-неравенств, образующих область допустимых значений.

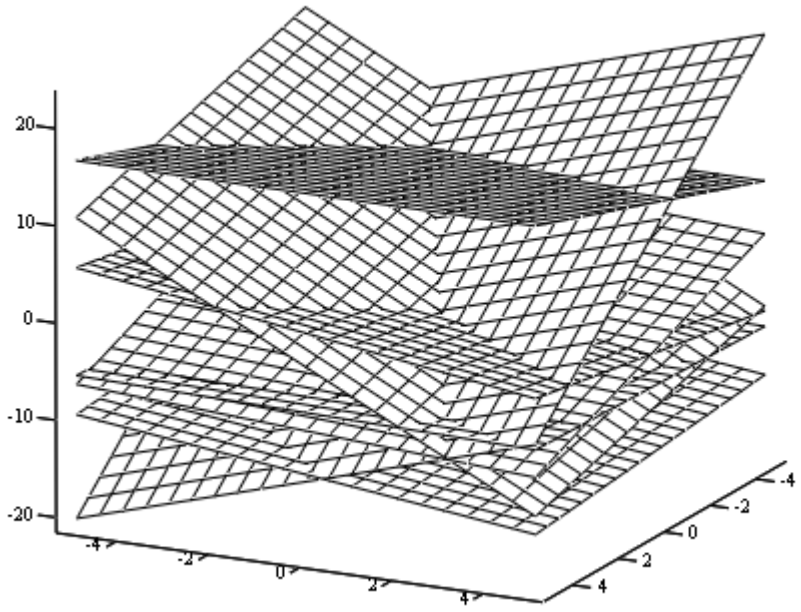

Рисунок 8.1 – Ограничения-неравенства

Каждая из поверхностей, изображённых на рисунке 8.1, представляет собой гиперплоскость, которая делит пространство на два полупространства, пересечения которых, в свою очередь, образуют многогранник допустимых значений.

Следующим шагом решения поставленной задачи является отыскание опорного решения, описанное в главе 5.

Введём начальную точку $x_0 = \begin{bmatrix} 0.1 \\ 0.3 \\ 0.5 \end{bmatrix}$.

Тогда следующей точкой x_1 будет являться проекция x_0 на ближайшую поверхность фигуры области допустимых значений с координатами:

$$x_1 = \begin{bmatrix} -0.0981658 \\ -0.216383 \\ 2.69785 \end{bmatrix}.$$

После итерационного перемещения по области допустимых значений рабочая точка будет иметь координаты одной из вершин, где хотя бы n ограничений-неравенств будут выполняться как строгие равенства. Таким образом, точка опорного решения:

$$x_k = \begin{bmatrix} 1.20628 \\ 0.357105 \\ 0.277923 \end{bmatrix}.$$

Завершающим шагом решения задачи является поиск точки, где значение целевого функционала $I(x)$ минимально (глава 6). Минимальным значением целевого функционала для тестового примера является $I(x) = -50.3752$.

Полученная точка оптимума имеет координаты:

$$x^* = \begin{bmatrix} 1. \\ 0.357105 \\ 0.277923 \end{bmatrix}.$$

Проверим корректность решения, выполненного посредством написанной программы путём подстановки полученной точки в преобразованный целевой функционал $I(x^*) = x^{*T} Q' x^* + p'^T x^*$.

Действительно, рассчитав значение целевого функционала в полученной точке оптимума, получим $I(x) = -50.3752$, что подтверждает правильность найденного посредством написанной программы решения поставленной задачи квадратичного программирования.

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8Б51	Кружков Денис Сергеевич

Школа	ИШИТР	Отделение (НОЦ)	Информационных технологий
Уровень образования	Бакалавриат	Направление/специальность	Прикладная математика и информатика

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	*Человеческие ресурсы – 2 человека (руководитель и студент-дипломник).*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Планирование и формирование бюджета научных исследований*	*Необходимо составить календарный план НИ и рассчитать затраты НИ.*
2. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования*	*Расчет интегральных показателей эффективности исследования, выбор наилучшего исполнения.*
Перечень графического материала:	
1. *Оценка конкурентоспособности технических решений* 2. *Матрица SWOT* 3. *График проведения и бюджет НИ* 4. *Оценка ресурсной, финансовой и экономической эффективности НИ*	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Подопригора Игнат Валерьевич	Кандидат экономических наук		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8Б51	Кружков Денис Сергеевич		

9 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

Целью написания раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» является проектирование и создание конкурентоспособных разработок, отвечающих современным требованиям в области ресурсоэффективности и ресурсосбережения.

9.1 Оценка коммерческого и инновационного потенциала научных исследований

9.1.1 Потенциальные потребители результатов исследований

Выполненная работа направлена на разработку и реализацию алгоритма, который будет решать задачи квадратичного программирования с использованием оператора-проектора.

Целевой рынок – сегменты рынка, на котором будет продаваться в будущем разработка. В свою очередь, сегмент рынка – это особым образом выделенная часть рынка, группы потребителей, обладающих определенными общими признаками.

Сегментирование – это разделение покупателей на однородные группы, для каждой из которых может потребоваться определенный товар (услуга). Можно применять географический, демографический, поведенческий и иные критерии сегментирования рынка потребителей, возможно применение их комбинаций с использованием таких характеристик, как возраст, пол, национальность, образование, любимые занятия, стиль жизни, социальная принадлежность, профессия, уровень дохода.

В зависимости от категории потребителей (коммерческие организации, физические лица) необходимо использовать соответствующие критерии сегментирования. Например, для коммерческих организаций критериями сегментирования могут быть: месторасположение; отрасль; выпускаемая продукция; размер и др.

Потенциальными потребителями результатов исследования являются крупные организации, специализирующиеся в нефтегазовой отрасли, а именно газодобывающие компании, которые имеют в своём распоряжении уже эксплуатируемые УКПГ или планируют внедрение новых установок подготовки газа. Для этих компаний разрабатывается автоматизированная система управления узла осушки газа установки комплексной подготовки газа.

9.1.2 Анализ конкурентных технических решений

Анализ конкурентных технических решений проводится с помощью оценочной карты для сравнения конкурентных технических решений, приведенной в таблице 9.1.2.1. Основным методом, использующимся в данной работе, является метод с использованием оператора-проектора, который используется в поиске оптимума при попадании в многогранник допустимых значений. Конкурентные методы: метод Баранкина-Дорфмана, метод Хилдрета-Д'Эзопо.

Таблица 9.1.2.1 – Оценочная карта для сравнения конкурентных технических решений

Критерии оценки	Вес	Баллы			Конкуренто-способность		
		$Б_{осн}$	$Б_{см}$	$Б_{пп}$	$Б_{осн}$	$Б_{см}$	$Б_{пп}$
Технические критерии оценки ресурсоэффективности							
Эффективность	0,3	4	4	4	1,2	1,2	1,2
Устойчивость	0,2	5	5	5	1	1	1
Временные затраты	0,2	5	4	4	1	0,8	0,8
Новизна метода	0,1	5	3	3	0,5	0,3	0,3
Простота реализации	0,1	5	4	5	0,5	0,4	0,5
Универсальность	0,1	4	4	4	0,5	0,4	0,4
Итого	1	29	24	25	4,7	4,1	4,2

По полученным результатам можно сделать вывод, что разрабатываемый алгоритм для оценки информативности является по конкурентоспособности наиболее эффективным.

9.1.3 SWOT-анализ

SWOT-анализ – это эффективный инструмент стратегического менеджмента. SWOT-анализ заключается в исследовании внешней и внутренней сред проекта. метод, которые заключается в комплексном анализе научно-исследовательского проекта. SWOT – Strengths (сильные стороны), Weaknesses (слабые стороны), Opportunities (возможности) и Threats (Угрозы).

Разработанная для алгоритма поиска информативного атрибута матрица SWOT представлена в таблице 9.1.3.1.

Таблица 9.1.3.1 – SWOT- анализ

	Сильные стороны: С1. Высокая эффективность алгоритма для решения задач ЛП. С2. Гибкость алгоритма с точки зрения считывания исходных данных. С3. Простая эксплуатация.	**Слабые стороны**: Сл1. Узкий круг целевой аудитории. Сл2. Трудоемкий процесс написания и отладки программы для решения задач ЛП.
Возможности: В1. Расширение функционала. В2. Написание алгоритма на других языках программирования.	Благодаря гибкости и высокой эффективности алгоритм может быть дополнен реализацией других задач, например, задач квадратичного программирования.	Написание алгоритма на другом языке программирования, а также расширение функционала могут сделать алгоритм более широко применимым, увеличить целевую аудиторию.
Угрозы: У1. Отсутствие спроса на продукт на рынке. У2. Развитие и появление аналогов алгоритма.	Наглядные результаты использования алгоритма, а также его низкая стоимость могу увеличить спрос на него.	Узкая направленность и использование платного языка программирования могут ослабить интерес покупателей.

9.2 Планирование научно-исследовательских работ

9.2.1 Структура работ в рамках научного исследования

Трудоемкость выпускной квалификационной работы определяется опытным путем в человеко-днях и имеет вероятностный характер, так как зависит от трудно учитываемых факторов. Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

- определение структуры работ в рамках научного исследования;
- определение участников каждой работы;
- установление продолжительности работ;
- построение графика проведения научных исследований.

В качестве исполнителей в проекте представлены руководитель и инженер. Перечень этапов, работ и исполнителей приведен в таблице 9.2.1.1.

Таблица 9.2.1.1 – Основные этапы выполнения ВКР

№	Описание этапа	Исполнители	Загруженность исполнителей
1	Составление и утверждение задания	НР	НР – 100%
2	Анализ предметной области	С, НР	С – 50%, НР – 50%
3	Разработка календарного плана	С, НР	С – 10%, НР – 90%
4	Обзор литературы и интернет - источников	С, НР	С – 50%, НР – 50%
5	Обзор задач и методов решения задач ЛП	С	С – 100%
6	Математическая постановка задачи	С, НР	С – 10%, НР – 90%
7	Выбор метода решения поставленной задачи	С, НР	С – 10%, НР – 90%

Таблица 9.2.1.1 (продолжение) – Основные этапы выполнения ВКР

8	Разработка алгоритмов решения задачи	С	С – 100%
9	Программная реализация компьютерной модели	С	С – 100%
10	Проведение исследований на модельных данных	С, НР	С – 80% НР – 20%
11	Проведение исследований на реальных данных	С, НР	С – 60% НР – 40%
12	Расчет экономической эффективности научно-технической продукции	С	С – 100%
13	Оценка социальной ответственности проекта	С	С – 100%
14	Написание пояснительной записки	С, НР	С – 90% НР – 10%

9.2.2 Определение трудоемкости выполнения работ

Трудовые затраты в большинстве случаев образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

Трудоемкость выполнения научного исследования оценивается экспертным путем в человеко-днях и носит вероятностный характер, т.к. зависит от множества трудно учитываемых факторов. Для определения ожидаемого (среднего) значения трудоемкости используется следующая формула:

$$t_{\text{ож}} = \frac{3 \cdot t_{min} + 2 \cdot t_{max}}{5}, \quad (9.2.2.1)$$

где t_{min} – предположительно минимальная продолжительность этапа в рабочих днях, определяемая методом экспертной оценки;

t_{max} – предположительно максимальная продолжительность этапа в рабочих днях, определяемая методом экспертной оценки.

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях $T_{\text{р}}$, учитывающая параллельность выполнения работ несколькими исполнителями.

$$T_{\text{р}_i} = \frac{t_{\text{ож}i}}{\text{Ч}_i}, \quad (9.2.2.2)$$

где $T_{\text{р}_i}$ – продолжительность одной работы, раб. дн.;

$t_{ожi}$ – ожидаемая трудоемкость выполнения одной работы, чел.-дн.

$Ч_i$ – численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

Продолжительность каждого этапа рассчитывается по формуле:

$$t_{раб} = t_{ож} \cdot k_д, \quad (9.2.2.3)$$

где $t_{раб}$ – длительность этапов в рабочих днях;

$k_д$ – коэффициент, учитывающий дополнительное время на консультации и согласование работ, $k_д$ = 1.2.

9.2.3 Разработка графика проведения научного исследования

В данном пункте необходимо построить диаграмму Ганта – горизонтально-ленточный график, на котором работы представляются протяженными во времени отрезками, характеризующимися датами начала и окончания выполнения приведенных работ.

Линейный график строится на основании полученных значений $t_{раб}$, предварительно переведенных в календарные дни по формуле:

$$t_к = t_{раб} \cdot K_н, \quad (9.2.3.1)$$

где $t_к$ – длительность этапов работ в календарных днях;

$K_н$ – коэффициент календарности.

Коэффициент календарности рассчитывается по формуле:

$$K_н = \frac{T_к}{T_к - T_{вд} - T_{пд}}, \quad (9.2.3.2)$$

где $T_к$ – количество календарных дней, $T_к$ = 365;

$T_{вд}$ – количество выходных дней, $T_{вд}$ 108;

$T_{пд}$ – количество праздничных дней, $T_{пд}$ =10;

$$K_н = \frac{365}{365-108-10} = 1.478 \quad (9.2.3.3)$$

Все расчеты сведены в таблицу 9.2.3.1.

Таблица 9.2.3.1 – Временные показатели проведенных работ

№	Исполнители	Продолжительность работ в днях			Трудоемкость	
		t_{min}	t_{max}	$t_{ож}$	$t_{раб}$	t_k
1	НР	1	2	1,4	1,68	2,48
2	С, НР	2	3	2,4	2,88	4,26
3	С, НР	1	2	1,4	1,68	2,48
4	С, НР	3	5	3,8	4,56	6,74
5	С	2	5	3,2	3,84	5,68
6	С, НР	3	6	4,2	5,04	7,45
7	С, НР	5	8	6,2	7,44	11,00
8	С	3	6	4,2	5,04	7,45
9	С	4	6	4,8	5,76	8,51
10	С, НР	2	5	3,2	3,84	5,68
11	С, НР	2	4	2,8	3,36	4,97
12	С	5	7	5,8	6,96	10,29
13	С	4	5	4,4	5,28	7,80
14	С, НР	9	12	10,2	12,24	18,09
Итого				58	69,6	102,87

На основании таблицы 6 построим диаграмму Ганта (рисунок 9.2.3.1).

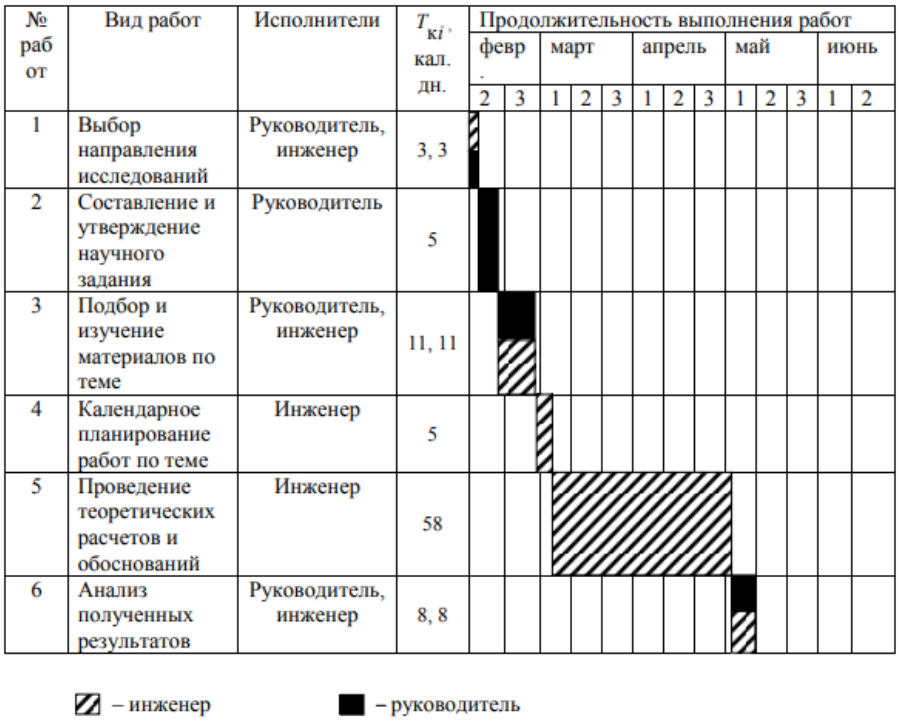

Рисунок 9.2.3.1 – Диаграмма Ганта

9.3 Бюджет научно-технического исследования (НТИ)

9.3.1 Расчет затрат на специальное оборудование для научных (экспериментальных) работ

В данную статью включают все затраты, связанные с приобретением специального оборудования, необходимого для проведения работ по конкретной теме.

В ходе выполнении НТИ использовалось имеющееся компьютерное оборудование, поэтому его стоимость учитывается в калькуляции в виде амортизационных отчислений за 5 месяцев (таблица 9.3.1.1).

Таблица 9.3.1.1 – Расчет бюджета затрат на приобретение спецоборудования для научных работ

Наименование	Единица измерения	Количество	Цена за единицу, руб.	Затраты на материалы, руб.
Амортизация оборудования	шт.	1	16000	16000
Итого				16

9.3.2 Основная заработная плата исполнителей темы

Основная заработная плата руководителя рассчитывается на основании отраслевой оплаты труда (оклад, стимулирующие выплаты, районный коэффициент). Величина расходов по заработной плате определяется исходя из трудоемкости выполняемых работ и действующей системы окладов и тарифных ставок. В состав основной заработной платы включается премия, выплачиваемая ежемесячно из фонда заработной платы в размере 20 –30 % от тарифа или оклада. Для студента-дипломника основную заработную плату составляет государственная стипендия с учетом районного коэффициента.

Для расчета основной заработной платы (таблица 9.4.2.2) необходимо привести действительный годовой фонд рабочего времени руководителя и студента (табл. 9.3.2.1).

Таблица 9.3.2.1 – Баланс времени

Показатели рабочего времени	Руководитель	Студент-дипломник
Календарное число дней	365	365
Количество нерабочих дней (выходные дни и праздничные дни)	52 и 10	108 и 10
Потери рабочего времени (отпуск, больничные), дни	48	24
Действительный годовой фонд рабочего времени, дни	255	223

Статья включает основную заработную плату работников, непосредственно занятых выполнением НТИ, и дополнительную заработную плату:

$$З_{зп} = З_{осн} \cdot З_{доп}, \quad (9.3.2.1)$$

где $З_{осн}$ – основная заработная плата;

$З_{доп}$ – дополнительная заработная плата (10% от основной).

Основная заработная плата руководителя (лаборанта, инженера) от предприятия рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot T_р \quad (9.3.2.2)$$

где $З_{осн}$ – основная заработная плата одного работника;

$З_{дн}$ – среднедневная заработная плата работника, руб.;

$T_р$ – продолжительность работ, выполняемых работником, раб.дн.

Среднедневная зарплата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot M}{F_д}, \quad (9.3.2.3)$$

где $З_м$ – месячный должностной оклад работника, руб.;

M – количество месяцев работы без отпуска в течение года:

при отпуске в 48 раб. дней М = 10,4 месяца, 6-дневная неделя;

$F_д$ – действительный годовой фонд рабочего времени научно-технического персонала, раб. дн., равный 251.

Месячный должностной оклад работника:

$$З_м = З_{тс} \cdot (1 + k_{пр} + k_д) \cdot k_р, \quad (9.3.2.4)$$

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_{пр}$ – премиальный коэффициент, равный 0,3;

$k_д$ – коэффициент доплат и надбавок составляет примерно 0,2-0,5;

$k_р$ – районный коэффициент, равный 1,3 для Томска.

Расчет основной платы представлен в таблице 9.3.2.2.

Таблица 9.3.2.2 – Расчет основной заработной платы

Исполнитель	Тарифная заработная плата, руб.	Районный коэффициент	Месячный должностной оклад работника, руб.	Среднедневная заработная плата, руб.	Продолжительность работ, дни	Заработная плата основная, руб.
Руководитель	23265	1,3	30244,5	1218	43	52374
Итого:						52374

9.3.3 Дополнительная заработная плата

Затраты по дополнительной заработной плате исполнителей темы учитывают величину предусмотренных Трудовым кодексом РФ доплат за отклонение от нормальных условий труда, а также выплат, связанных с обеспечением гарантий и компенсаций (при исполнении государственных и общественных обязанностей, при совмещении работы с обучением, при предоставлении ежегодного оплачиваемого отпуска и т.д.).

Расчет дополнительной заработной платы ведется по следующей формуле:

$$З_{доп} = k_{доп} \cdot З_{осн} = 0.15 \cdot 52374 = 7856.1 \quad (9.3.3.1)$$

где $З_{осн}$ – основная заработная плата;

$k_{доп}$ – коэффициент дополнительной заработной платы на стадии проектирования, принимается равным 0,15.

9.3.4 Отчисления во внебюджетные фонды (страховые отчисления)

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органами государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

Величина отчислений во внебюджетные фонды определяется исходя из следующей формулы:

$$З_{внеб} = k_{внеб} \cdot (З_{осн} + З_{доп}), \quad (9.3.4.1)$$

где $k_{внеб}$ – коэффициент отчислений на уплату во внебюджетные фонды (пенсионный фонд, фонд обязательного медицинского страхования и пр.).

Для учреждений, осуществляющих образовательную и научную деятельность в 2019г., водится пониженная ставка 28% (п. 6 ч. 1 ст. 58 Закона 212-ФЗ).

Расчет отчислений во внебюджетные фонды приведен в таблице 9.3.4.1.

Таблица 9.3.4.1 – Отчисления во внебюджетные фонды

Исполнитель	Основная заработная плата, руб.	Дополнительная заработная плата, руб.
Руководитель проекта	52374	7856,1
Коэффициент отчисления во внебюджетные фонды	14664,72	2199,7
Итого		16864,42

9.3.5 Прочие прямые затраты

К данному виду затрат относятся затраты на электроэнергию. Для юридических лиц стоимость 1 кВт·ч составляет 5,8 рублей. При умеренном пользовании ноутбук средней мощности потребляет 100 Вт в час в среднем. В день на работу затрачивается 6 часов, всего на работу с компьютером и оборудованием затрачивается 34 дня у инженера и 9 дней у руководителя. Тогда затраты на электроэнергию составят:

$$З_{эн} = 100 \cdot \frac{5,8}{1000} \cdot 6 \cdot 43 = 149.64 \text{ руб.} \quad (9.3.5.1)$$

9.3.6 Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов: печать и ксерокопирование материалов исследования, оплата услуг связи, электроэнергии, почтовые и телеграфные расходы, размножение материалов и т.д. Их величина определяется по следующей формуле:

$$З_{накл} = k_{накл} \cdot (З_{осн} + З_{доп}), \quad (9.3.6.1)$$

где $k_{накл}$ – коэффициент накладных расходов, 16 %.

Получим:

$$З_{накл} = 0{,}16 \cdot (48\,681.98 + 5\,841.84) = 8723.81 \text{ руб.}$$

9.3.7 Формирование бюджета затрат научно-исследовательского проекта

Рассчитанная величина затрат научно-исследовательской работы является основой для формирования бюджета затрат проекта, который при формировании договора с заказчиком защищается научной организацией в качестве нижнего предела затрат на разработку научно-технической продукции.

Определение бюджета затрат на научно-исследовательский проект приведено в таблице 9.4.7.1.

Таблица 9.3.7.1 – Расчет бюджета затрат

Наименование статьи	**Сумма, руб.**
1. Затраты на специальное оборудование	16 000
2. Затраты по основной заработной плате исполнителей проекта	52 374
3. Затраты по дополнительной заработной плате исполнителей проекта	7856
4. Отчисления во внебюджетные фонды	16864
5. Накладные расходы	14 895
Бюджет затрат НТИ	107 989

9.4 Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности происходит на основе расчета интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

Интегральный финансовый показатель разработки определяется как:

$$I_{\text{фин}_\text{р}}^{\text{исп}_i} = \frac{\Phi_{p_i}}{\Phi_{max}}, \qquad (9.4.1)$$

где $I_{\text{фин}_\text{р}}^{\text{исп}_i}$ – интегральный финансовый показатель разработки;

Φ_{p_i} – стоимость i-го варианта исполнения;

Φ_{max} – максимальная стоимость исполнения научно-исследовательского проекта.

Φ_{max} зависит от сложности проекта для которого разрабатывается ПО.

Т.к. стоимость всех вариантов исполнения одинакова, интегральные финансовые показатели также будут одинаковы и равны 1.

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить следующим образом:

$$I_{p_i} = \sum a_i \cdot b_i, \qquad (9.4.2)$$

где I_{p_i} – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки;

a_i – весовой коэффициент i-го варианта исполнения разработки;

b_i^a, b_i^p – бальная оценка i-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания.

Интегральный показатель эффективности вариантов исполнения разработки определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формулам:

$$I_{\text{исп}_1} = \frac{I_{p-\text{исп}_1}}{I_{\text{фин}_p}^{\text{исп}_1}}$$
$$I_{\text{исп}_2} = \frac{I_{p-\text{исп}_2}}{I_{\text{фин}_p}^{\text{исп}_2}} \quad (9.4.3)$$

Так как интегральные финансовые показатели одинаковы и равны 1, то интегральные показатели эффективности вариантов исполнения разработки равны соответствующим интегральным показателям ресурсоэффективности.

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволит определить сравнительную эффективность проекта и выбрать наиболее целесообразный вариант из предложенных.

Сравнительная эффективность проекта:

$$\text{Э}_{\text{ср}} = \frac{I_{\text{исп}_i}}{I_{\text{исп}_1}} \quad (9.4.4)$$

В пункте 9.2 было рассмотрено два варианта исполнения алгоритма. На основании этого необходимо провести сравнительную характеристику вариантов исполнения (таблица 9.5.1).

Таблица 9.5.1– Сравнительная оценка характеристик вариантов исполнения проекта

Критерии	Весовой коэффициент	Исп.1	Исп.2
1. Повышение производительности труда	0,3	5	5
2. Удобство в эксплуатации	0,2	4	3
3. Удобство в считывании исходных данных	0,2	5	2
4. Скорость работы	0,1	4	3
5. Простота эксплуатации	0,1	4	3
6. Техническая поддержка платформы	0,1	5	3
I_{p_i}		4,6	3,4

На основании полученных показателей выполним сравнение интегрального показателя эффективности вариантов исполнения разработки (табл. 9.4.2).

Таблица 9.4.2 – Сравнительная эффективность разработки

Показатели	Исп.1	Исп.2
Интегральный финансовый показатель разработки	1	1
Интегральный показатель ресурсоэффективности разработки	4,6	3,4
Интегральный показатель эффективности	4,6	3,4
Сравнительная эффективность вариантов исполнения	1	0,74

С позиции финансовой и ресурсной эффективности на основании таблицы, первый вариант исполнения системы наиболее выгодный. Данный вариант исполнения и используется в выпускной квалификационной работе.

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8Б51	Кружков Денис Сергеевич

Школа	ИШИТР	**Отделение (НОЦ)**	Информационных технологий
Уровень образования	Бакалавриат	**Направление/специальность**	Прикладная математика и информатика

Тема ВКР:

«Разработка и реализация алгоритма решения задач квадратичного программирования на основе оператора-проектора»	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения.	1. Реализовать алгоритм решения задач квадратичного программирования на основе оператора-проектора. Область применения: задачи о смесях, транспортные задачи.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности. 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности.	1.1 Рассмотрены вредные факторы: Отклонение показателей микроклимата рабочей зоны; повышенный уровень шума на рабочем месте; отсутствие или недостаток естественного света, а также недостаточная освещенность рабочей зоны. 1.2 Рассмотрены опасные факторы: повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека; возгорание эксплуатируемого оборудования.
2. Экологическая безопасность: 2.1 Анализ воздействия объекта на окружающую среду; 2.2 Разработать решения по обеспечению экологической безопасности со ссылками на НТД по охране окружающей среды.	2.1 Рассмотрены негативно влияющие на экологию факторы при эксплуатации компьютера. 2.2 Решения по обеспечению экологической безопасности согласно нормативным документам.
3. Безопасность в чрезвычайных ситуациях: 3.1 Перечень возможных ЧС при разработке и эксплуатации проектируемого решения; 3.2 Разработка действий в результате возникшей ЧС и мер по ликвидации её последствий.	3.1 Перечень возможных ЧС, которые могут возникнуть при работе в помещении офиса. 3.2 Способы защиты от пожара и ликвидация последствий.
4. Правовые и организационные вопросы обеспечения безопасности: 4.1 Организационные мероприятия при компоновке рабочей зоны.	4.1 Организационные мероприятия по обеспечению безопасности трудящихся за персональным компьютером.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент ООТД	Мезенцева Ирина Леонидовна			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8Б51	Кружков Денис Сергеевич		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8Б51	Кружков Денис Сергеевич

Школа	ИШИТР	**Отделение (НОЦ)**	Информационных технологий
Уровень образования	Бакалавриат	**Направление/специальность**	Прикладная математика и информатика

Тема ВКР:

«Разработка и реализация алгоритма решения задач квадратичного программирования на основе оператора-проектора»	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения.	1. Алгоритм решения задач квадратичного программирования на основе оператора-проектора. Область применения: задачи о смесях, транспортные задачи.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности. 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности.	1.2 Рассмотрены вредные факторы: Отклонение показателей микроклимата рабочей зоны; повышенный уровень шума на рабочем месте; отсутствие или недостаток естественного света, а также недостаточная освещенность рабочей зоны. 1.2 Рассмотрены опасные факторы: повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека; возгорание эксплуатируемого оборудования.
2. Экологическая безопасность: 2.3 Анализ воздействия объекта на окружающую среду; 2.4 Разработать решения по обеспечению экологической безопасности со ссылками на НТД по охране окружающей среды.	2.1 Рассмотрены негативно влияющие на экологию факторы при эксплуатации компьютера. 2.2 Решения по обеспечению экологической безопасности согласно нормативным документам.
3. Безопасность в чрезвычайных ситуациях: 3.3 Перечень возможных ЧС при разработке и эксплуатации проектируемого решения; 3.4 Разработка действий в результате возникшей ЧС и мер по ликвидации её последствий.	3.1 Перечень возможных ЧС, которые могут возникнуть при работе в помещении офиса. 3.2 Способы защиты от пожара и ликвидация последствий.
4. Правовые и организационные вопросы обеспечения безопасности: 4.2 Организационные мероприятия при компоновке рабочей зоны.	4.1 Организационные мероприятия по обеспечению безопасности трудящихся за персональным компьютером.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент ООТД	Мезенцева Ирина Леонидовна			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8Б51	Кружков Денис Сергеевич		

10 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Введение

Выпускная квалификационная работа представляет собой сравнительное исследование моделей параметризации результатов тестирования. Во время выполнения данной работы были разработаны и запрограммированы алгоритмы реализации параметрических моделей для оценки учебных достижений студентов посредством тестов. Все расчеты и алгоритмы реализованы в среде Visual Studio.

Данная работа имеет высокий потенциал. Описанный в работе алгоритм может использоваться для, например, решения транспортных задач, задач о смесях и других задач линейного программирования. Данный способ имеет ряд преимуществ.

Предполагаемое место работы – компьютерный класс Кибернетического центра ТПУ. Основные средства работы – персональный компьютер и локальная вычислительная сеть с выходом в Интернет. Все действия, описанные в данной выпускной квалификационной работе, были совершены в городе Томск, на территории кампуса томского политехнического университета.

В разделе будут рассмотрены опасные и вредные факторы, оказывающие влияние на производственную деятельность технологического персонала, работающего с автоматизированной системой управления технологическим процессом, рассмотрены воздействия разрабатываемой системы на окружающую среду, правовые и организационные вопросы, а также мероприятия в чрезвычайных ситуациях.

10.1 Правовые и организационные вопросы обеспечения безопасности

10.1.1 Специальные правовые нормы трудового законодательства

Как уже было неоднократно отмечено, при работе с персональным компьютером очень важную роль играет соблюдение правильного режима труда

и отдыха. В противном случае у персонала отмечаются значительное напряжение зрительного аппарата с появлением жалоб на неудовлетворенность работой, головные боли, раздражительность, нарушение сна, усталость и болезненные ощущения в глазах, в пояснице, в области шеи и руках.

При восьмичасовой рабочей смене на ВДТ и ПЭВМ перерывы в работе должны составлять от 10 до 20 минут каждые два часа работы. В перерывах, рекомендуется проводить комплекс упражнений для глаз.

По типовой инструкции по охране труда при работе на персональном компьютере ТОИ Р-45-084-01 всё же вменяется установление перерывов. В целом продолжительность непрерывной работы за компьютером не должна превышать 2-х часов.

Следует обратить внимание на то, что основная работа за компьютером предусматривает не менее 50 % времени в течение рабочей смены или рабочего дня нахождения за ним. Время перерыва зависит от вида и сложности осуществляемой работы путем деления на группы. Выделяют 3 группы: А (работа по считыванию информации с экрана компьютера с предварительным запросом), Б (работа по вводу информации), В (творческая работа в режиме диалога с компьютером).

В зависимости от сложности работы, установление числа и длительности перерывов происходит следующим образом:

1) для группы А (не свыше 60000 считываемых знаков за смену) перерыв составляет 15 минут, предоставляется два раза – через два часа после начала работы и перерыва на обед;

2) для группы Б (не свыше 40000 вводимых знаков за смену) перерыв составляет 10 минут через каждый трудовой час;

3) для группы В (не свыше шести 6 часов за смену) перерыв составляет 15 минут через каждый трудовой час.

Согласно ТК РФ Статья 135 об установлении заработной платы, заработная плата работнику устанавливается трудовым договором в соответствии с действующими у данного работодателя системами оплаты труда.

Что же касается нормы выработки, времени, нормативы численности и других норм – они устанавливаются в соответствии с достигнутым уровнем техники, технологии, организации производства и труда. Нормы труда могут быть пересмотрены по мере совершенствования или внедрения новой техники, технологии и проведения организационных либо иных мероприятий, обеспечивающих рост производительности труда, а также в случае использования физически и морально устаревшего оборудования. Достижение высокого уровня выработки продукции (оказания услуг) отдельными работниками за счет применения по их инициативе новых приемов труда и совершенствования рабочих мест не является основанием для пересмотра ранее установленных норм труда.

Конкретные размеры повышения оплаты труда устанавливаются работодателем с учетом мнения представительного органа работников в порядке, установленном статьей 372 настоящего Кодекса для принятия локальных нормативных актов, либо коллективным договором, трудовым договором.

10.1.2 Организационные мероприятия при компоновке рабочей зоны

Под проектированием рабочего места понимается целесообразное пространственное размещение в горизонтальной и вертикальной плоскостях функционально взаимоувязанных средств производства (оборудования, оснастки, предметов труда и др.), необходимых для осуществления трудового процесса. При проектировании рабочих мест должны быть учтены освещенность, температура, влажность, давление, шум, наличие вредных веществ, электромагнитных полей и другие санитарно-гигиенические требования к организации рабочих мест.

Площадь одного рабочего места с компьютером должна быть не менее 6 $м^2$. При размещении рабочих мест с персональными компьютерами должны учитываться расстояния между рабочими столами с мониторами.

Для комфортной работы стол инженера-программиста должен удовлетворять следующим условиям:

1) высота стола должна быть выбрана с учетом возможности сидеть свободно, в удобной позе, при необходимости опираясь на подлокотники;

2) нижняя часть стола должна быть сконструирована так, чтобы программист мог удобно сидеть, не был вынужден поджимать ноги;

3) поверхность стола должна обладать свойствами, исключающими появление бликов в поле зрения программиста;

4) конструкция стола должна предусматривать наличие выдвижных ящиков.

5) высота рабочей поверхности рекомендуется в пределах 680-760мм.

6) Высота поверхности, на которую устанавливается клавиатура, должна быть около 650мм.

Помещения с компьютерами в обязательном порядке должны быть оборудованы системами эффективной приточно-вытяжной вентиляцией отопления и кондиционирования воздуха. Внутренняя отделка интерьера помещений с компьютерами должна быть сделана при использовании диффузно-отражающих материалов. В обязательном порядке в помещении должны находиться углекислотный огнетушитель для тушения пожара и аптечка первой медицинской помощи.

Создание благоприятных условий труда и правильное эстетическое оформление рабочих мест на производстве имеет большое значение, как для облегчения труда, так и для повышения его привлекательности, положительно влияющей на производительность труда.

Рабочее место в комнате № 104 КЦ ТПУ отвечает данным условиям.

10.2 Профессиональная социальная безопасность

В данном подразделе производится анализ вредных и опасных факторов, которые могут возникнуть на рабочем месте при проведении исследований и использовании результатов исследования.

Опасным производственным фактором (ОПФ) называется такой производственный фактор, воздействие которого на работающего мгновенно приводит к травме или летальному исходу. Травма – это повреждение тканей организма и нарушение его функций внешним воздействием.

Вредным производственным фактором (ВПФ) называется такой производственный фактор, воздействие которого на работающего в определенных условиях приводит к заболеванию или снижению трудоспособности.

Выявленные факторы представлены в таблице 10.2.1

Таблица 10.2.1 – Опасные и вредные факторы при проведении сравнительного исследования результатов оценивания учебных достижений

Источник фактора, наименование видов работ	Факторы		Нормативные документы
	Вредные	Опасные	
– Работа за персональной электронно-вычисли-тельной машиной в компьютерной аудитории.	– Отклонение показателей микроклимата рабочей зоны; – отсутствие или недостаток естественного света, а также недостаточная освещенность рабочей зоны; – повышенный уровень шума на рабочем месте;	– Повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека; возгорание эксплуатируемого оборудования.	–Гигиенические требования к микроклимату производственных помещений, СанПиН –2.2.4-548-96; –Гигиенические требования к естественному, искусственному и совмещённому освещению жилых и общественных зданий, СанПиН 2.2.1/2.1.1.1278–03 –Шум. Общие требования безопасности, СН 2.2.4/2.1.8.562–96; –Гигиенические требования к персональным электронно-вычислительным машинам и организации работы, СанПиН 2.2.2/2.4.1340-03; –Электробезопасность, ГОСТ 12.1.038–82 ССБТ

10.2.1 Анализ вредных и опасных факторов

10.2.1.1 Повышенный уровень шума на рабочем месте

Шум – колебания различной физической природы, отличающиеся сложностью спектральной и временной структуры. Шум создает значительную

нагрузку на нервную систему человека, оказывая на него психологическое воздействие.

Уровень звука на рабочих местах, связанных с творческой деятельностью, научной деятельностью, программированием, преподаванием и обучением не должен превышать 50 дБА согласно СН 2.2.4/2.1.8.562–96 [1].

Меры, которые необходимо принять, для того чтобы помещение было менее зашумленным – это обеспечить нормальную вентиляцию системного блока. Для охлаждения необходимо оборудовать со стороны вентиляционных отверстий хотя бы 20-30 см свободного пространства.

10.2.1.2 Отклонение показателей микроклимата рабочей зоны

Как центральное отопление, так и обогреватели сушат воздух. Пересушенный воздух при критических показателях создает неоспоримую опасность для здоровья человека: способствует возникновению инфекций, обострение аллергических заболеваний и астмы.

В помещениях, где установлены компьютеры, должны соблюдаться определенные параметры микроклимата. Эти нормы устанавливаются в зависимости от времени года, характера трудового процесса и характера производственного помещения. Оптимальные параметры микроклимата и нормы подачи свежего воздуха в помещения приведены на рисунках 10.2.1.2.1 и 10.2.1.2.2 (СанПиН 2.2.548-96) [2].

Период года	Категория работ по уровням энергозатрат, Вт	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Ia (до 139)	22 - 24	21 - 25	60 - 40	0,1
	Iб (140 - 174)	21 - 23	20 - 24	60 - 40	0,1
	IIa (175 - 232)	19 - 21	18 - 22	60 - 40	0,2
	IIб (233 - 290)	17 - 19	16 - 20	60 - 40	0,2
	III (более 290)	16 - 18	15 - 19	60 - 40	0,3
Теплый	Ia (до 139)	23 - 25	22 - 26	60 - 40	0,1
	Iб (140 - 174)	22 - 24	21 - 25	60 - 40	0,1
	IIa (175 - 232)	20 - 22	19 - 23	60 - 40	0,2
	IIб (233 - 290)	19 - 21	18 - 22	60 - 40	0,2
	III (более 290)	18 - 20	17 - 21	60 - 40	0,3

Рисунок 10.2.1.2.1 оптимальные величины показателей микроклимата на рабочих местах производственных помещений

Период года	Категория работ по уровню энерготрат, Вт	Температура воздуха, °С		Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с	
		диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более <**>
Холодный	Ia (до 139)	20,0 - 21,9	24,1 - 25,0	19,0 - 26,0	15 - 75 <*>	0,1	0,1
	Iб (140 - 174)	19,0 - 20,9	23,1 - 24,0	18,0 - 25,0	15 - 75	0,1	0,2
	IIa (175 - 232)	17,0 - 18,9	21,1 - 23,0	16,0 - 24,0	15 - 75	0,1	0,3
	IIб (233 - 290)	15,0 - 16,9	19,1 - 22,0	14,0 - 23,0	15 - 75	0,2	0,4
	III (более 290)	13,0 - 15,9	18,1 - 21,0	12,0 - 22,0	15 - 75	0,2	0,4
Теплый	Ia (до 139)	21,0 - 22,9	25,1 - 28,0	20,0 - 29,0	15 - 75 <*>	0,1	0,2
	Iб (140 - 174)	20,0 - 21,9	24,1 - 28,0	19,0 - 29,0	15 - 75 <*>	0,1	0,3
	IIa (175 - 232)	18,0 - 19,9	22,1 - 27,0	17,0 - 28,0	15 - 75 <*>	0,1	0,4
	IIб (233 - 290)	16,0 - 18,9	21,1 - 27,0	15,0 - 28,0	15 - 75 <*>	0,2	0,5
	III (более 290)	15,0 - 17,9	20,1 - 26,0	14,0 - 27,0	15 - 75 <*>	0,2	0,5

<*> При температурах воздуха 25 °С и выше максимальные величины относительной влажности воздуха должны приниматься в соответствии с требованиями п. 6.5.

<**> При температурах воздуха 26 - 28 °С скорость движения воздуха в теплый период года должна приниматься в соответствии с требованиями п. 6.6.

Рисунок 10.2.1.2.2 допустимые величины показателей микроклимата на рабочих местах производственных помещений

10.2.1.3 Отсутствие или недостаток естественного света, а также недостаточная освещенность рабочей зоны

Соответствующее производственное освещение способствует улучшению условий зрительной работы, как следствие снижает утомляемость и способствует повышению производительности труда.

Существует три вида освещения – естественное солнечное освещение, искусственное и комбинированное.

В компьютерных залах должно быть естественное и искусственное освещение, то есть комбинированное. Естественное освещение создается прямыми солнечными лучами. Искусственное освещение в помещениях эксплуатации компьютеров должно осуществляться системой общего равномерного освещения. Местное освещение не должно создавать бликов на поверхности экрана и увеличивать освещенность экрана более 300 лк согласно СНиП 23-05-95* [5].

Для оценки использования естественного света введено понятие коэффициента естественной освещенности (КЕО) и установлены минимальные допустимые значения КЕО — это отношение освещенности Ев внутри помещения за счет естественного света к наружной освещенности Ен от всей полусферы небосклона, выраженное в процентах:

$$КЕО = (\frac{E_в}{E_н}) 100\%.$$

По характеристике зрительской работы труд учащихся можно отнести ко второму разряду работы, и при боковом естественном освещении в аудитории, лаборатории на рабочих столах и партах должен обеспечиваться КЕО = 1,5 %. К средствам нормализации освещения производственных помещений и рабочих мест относятся: источники света, осветительные приборы, световые проемы, светозащитные устройства, светофильтры

10.2.1.4 Электрический ток

В связи с тем, что для работы компьютера используется электрическая энергия, его эксплуатация должна соответствовать правилам техники

безопасности при эксплуатации электроустановок потребителей. С этой точки зрения компьютер является потенциальным источником опасности поражения человека электрическим током.

Проходя через организм человека, электрический ток оказывает термическое, электролитическое и биологическое действие. Первое заключается в нагреве и ожогах различных участков тела человека, второе — в изменении состава и свойств крови.

Чтобы избежать поражения электрическим током, необходимо выполнять правила, указанные в [6].

10.2.2 Обоснование мероприятий по защите исследователя от действия опасных и вредных факторов

Организационные мероприятия при компоновке рабочей зоны

Большое значение для профилактики статических физических перегрузок имеет правильная организация рабочего места человека, работающего с ПЭВМ. Рабочее место должно быть организовано в соответствии с требованиями стандартов, технических условий и (или) методических указаний по безопасности труда. Оно должно удовлетворять следующим требованиям:

1) обеспечивать возможность удобного выполнения работ;
2) учитывать физическую тяжесть работ;
3) учитывать размеры рабочей зоны и необходимость передвижения в ней работающего;
4) учитывать технологические особенности процесса выполнения работ.

Невыполнение требований к расположению и компоновке рабочего места может привести к получению работником производственной травмы или развития у него профессионального заболевания. Рабочее место программиста должно соответствовать требованиям СанПин 2.2.2/2.4.1340-03.

Конструкция оборудования и рабочего места при выполнении работ в положении сидя должна обеспечивать оптимальное положение работающего,

которое достигается регулированием высоты рабочей поверхности, высоты сидения, оборудованием пространства для размещения ног и высотой подставки для ног. Схемы размещения рабочих мест с персональными компьютерами должны учитывать расстояния между рабочими столами с мониторами: расстояние между боковыми поверхностями мониторов не менее 1,2 м, а расстояние между экраном монитора и тыльной частью другого монитора не менее 2,0 м. Клавиатура должна располагаться на поверхности стола на расстоянии 100-300 мм от края, обращенного к пользователю. Быстрое и точное считывание информации обеспечивается при расположении плоскости экрана ниже уровня глаз пользователя, предпочтительно перпендикулярно к нормальной линии взгляда (нормальная линия взгляда 15 градусов вниз от горизонтали). Рабочие места с ПЭВМ при выполнении творческой работы, требующей значительного умственного напряжения или высокой концентрации внимания, рекомендуется изолировать друг от друга перегородками высотой 1,5 - 2,0 м.

Разрабатываемый в ходе выполнения ВКР программный продукт используется для анализа численных данных и наборов значений. При работе с данным продуктом не требуется постоянное его использование, так как программа анализирует входные данные и получает необходимый результат, который, непосредственно, будет использован в дальнейшем. Работнику необходимо контролировать параметры входных данных и фиксировать выходные параметры. Следовательно, преимуществом данного продукта является практически полная автономность.

10.3 Экологическая безопасность

Рассмотрим загрязнения литосферы в результате исследовательской деятельности бытовым мусором, на примере люминесцентных ламп. Их эксплуатация требует осторожности и четкого выполнения инструкции по обращению с данным отходом (код отхода 35330100 13 01 1, класс опасности – 1). В данной лампе содержится опасное вещество ртуть в газообразном

состоянии. При не правильной утилизации, лампа может разбиться и пары ртути могут попасть в окружающую среду. Вдыхание паров ртути может привести к тяжелому повреждению здоровья.

При перегорании ртутьсодержащей лампы (выходе из строя) её замену осуществляет лицо, ответственное за сбор и хранение ламп (обученное по электробезопасности и правилам обращения с отходом). Отработанные люминесцентные лампы сдаются только на полигон токсичных отходов для захоронения. Запрещается сваливать отработанные люминесцентные лампы с мусором.

Бытовой мусор помещений организаций несортированный, образованный в результате деятельности работников предприятия (код отхода 91200400 01 00 4). Агрегатное состояние отхода твердое; основные компоненты: бумага и древесина, металлы, пластмассы и др. Для сбора мусора рабочее место оснащается урной. При заполнении урны, мусор выносится в контейнер бытовых отходов. Предприятие заключает договор с коммунальным хозяйством по вывозу и размещению мусора на организованных свалках.

10.4 Безопасность в чрезвычайных ситуациях

10.4.1 Анализ вероятных ЧС

Наиболее вероятной ЧС в рамках рассматриваемого помещения является пожар. Помещение, в котором велась работа по степени пожаробезопасности относится к категории Г (умеренная пожароопасность), т.е. к помещению, в котором находятся негорючие вещества и материалы в горячем, раскаленном или расплавленном состоянии, процесс обработки которых сопровождается выделением лучистого тепла, искр и пламени.

Гипотетически возникновение пожара может возникнуть от следующих источников воспламенения [7]:

1) искра при разряде статистического электричества;
2) искра от электрооборудования;
3) открытое пламя.

Также на рабочем месте запрещается иметь огнеопасные вещества и выполнять следующее действия [7]:

1) курить;
2) зажигать огонь;
3) включать электрооборудование, если в помещении пахнет газом;
4) сушить что-либо на отопительных приборах;
5) закрывать вентиляционные отверстия в электроаппаратуре.

10.4.2 Обоснование мероприятий по предотвращению ЧС и разработка порядка действия в случае возникновения ЧС

К мерам по предупреждению пожара отнесем следующие пожарно-профилактические мероприятия:

1) соблюдение эксплуатационных норм оборудования;
2) обучение персонала правилам техники безопасности;
3) издание противопожарных инструкций, планов эвакуации.

Основными мерами по повышению устойчивости помещения к данной ЧС являются в первую очередь исключение образования благоприятной для пожара среды (контроль воздухообмена), а также использование трудно сгораемых материалов при отделке рабочего помещения [7].

Необходимо предусмотреть безопасную эвакуацию людей на случай возникновения пожара. При пожаре люди должны покинуть помещение в течение минимального времени. Помещение, в котором выполнялась работа,

входит в общий план эвакуации этажа, который предусматривает выход из всех помещений этажа в основной или запасной эвакуационные выходы здания. Эвакуация проводится согласно плану эвакуации, который выставлен на всеобщее обозрение в нескольких местах на каждом этаже.

В каждом кабинете установлен углекислотный огнетушитель ОУ-2 и табличка с указанием лица, ответственного за пожарную безопасность.

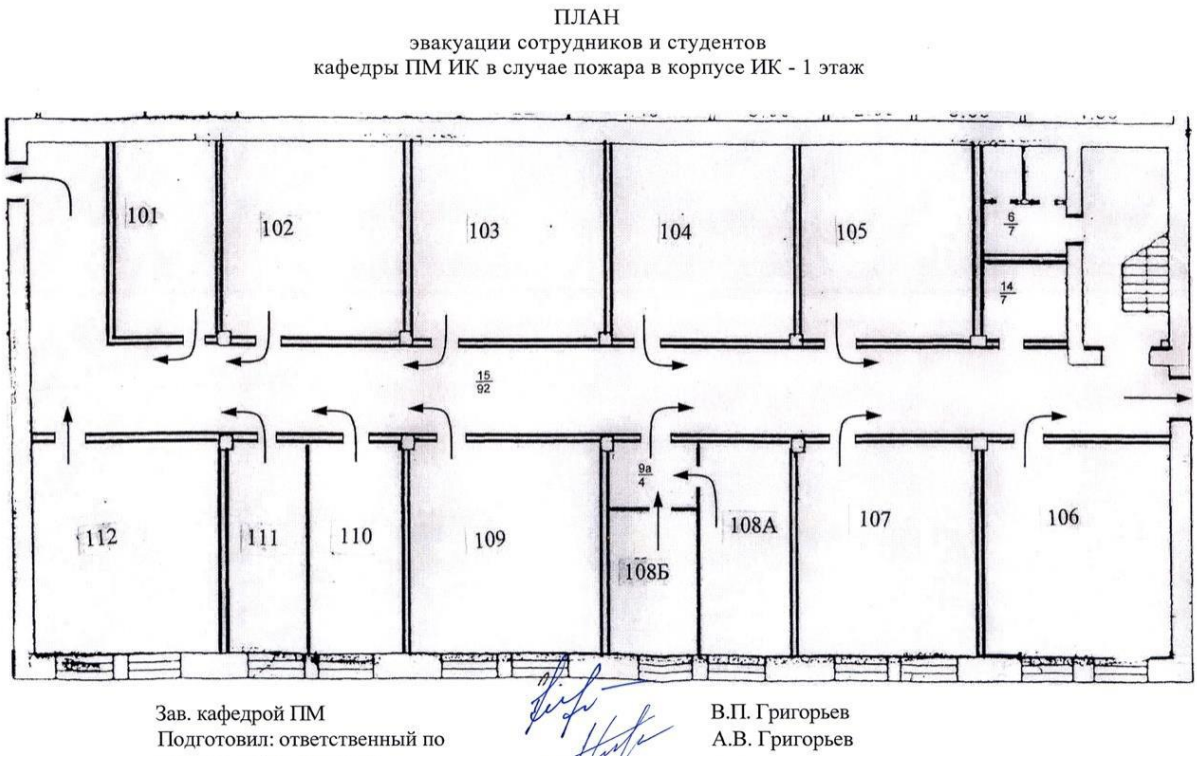

Рисунок 10.4.2.1 – План эвакуации

Необходимыми действиями в результате возникшей ЧС и мерами по ликвидации её последствий являются:

1. Передать сигнал «Тревога» голосом, задействовать систему оповещения людей о пожаре.
2. Сообщить по телефону 01, с сотового 010 адрес объекта, место возникновения пожара, свою фамилию. Сообщить по телефону 03, с сотового 030 адрес объекта, что случилось, информацию о пострадавших, свою фамилию, оказать помощь пострадавшим.
3. Открыть все эвакуационные выходы, направить людей к эвакуационным выходам согласно знакам направления движения.
4. Отключить от электропитания оборудование, механизмы и т.п., обесточить помещение.
5. По возможности принять меры по тушению пожара используя средства противопожарной защиты.

По возможности предотвратить развитие аварии, обозначить место аварии.

Выводы и рекомендации

Проанализировав условия труда на рабочем месте, где была разработана бакалаврская работа, можно сделать вывод, что помещение удовлетворяет необходимым нормам и в случае соблюдения техники безопасности и правил пользования компьютером работа в данном помещении не приведет к ухудшению здоровья работника. Само помещение и рабочее место в нем удовлетворяет всем нормативным требованиям. Кроме того, действие вредных и опасных факторов сведено к минимуму, т.е. микроклимат, освещение и электробезопасность соответствуют требованиям, предъявленным в соответствующих нормативных документах.

Работа с ЭВМ можно отнести к экологически безопасным видам деятельности, если должным образом утилизировать отходы данной деятельности.

Относительно рассмотренного вопроса об экологической безопасности можно сказать, что деятельность в ходе выполнения выпускной квалификационной работы не представляет опасности окружающей среды.

ЗАКЛЮЧЕНИЕ

В ходе выполнения преддипломной практики были получены следующие результаты:

– проведён обзор литературы в области линейного и квадратичного программирования;

– сформулирована актуальность решаемой задачи;

– сформулированы концептуальная и математическая постановки задачи;

– сформулированы выражения, позволяющие перейти от задачи квадратичного программирования со смешанными ограничениями к задаче только с ограничениями-неравенствами;

– разработана модель поиска решения задачи квадратичного программирования для заданного целевого функционала с наложенными ограничениями;

– составлены алгоритм преобразования целевого функционала с ограничений-равенств, а также алгоритм поиска искомого решения поставленной задачи;

– разработана программа на объектно-ориентированном языке C++ в среде Visual Studio, реализующая составленные алгоритмы;

– проверена и подтверждена работоспособность программы на тестовом примере.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Кюнци Г. П. Нелинейное программирование / Г. П. Кюнци, В. Крелле. – М.: Изд-во «Советское радио», 1965. – 303 с.
2. Вылегжанин О.Н., Шкатова Г.И. Учет ограничений равенств при решении оптимизационных задач с линейными ограничениями // Известия Томского политехнического университета. – 2008. – Т. 312. – № 5. – С. 76–78.
3. Вылегжанин О.Н., Шкатова Г.И. Решение задачи линейного программирования с использованием оператора-проектора // Известия Томского политехнического университета. – 2009. – Т. 314. – № 5. – С. 37–40.
4. Гилл Ф., Мюррей У., Райт М. Практическая Оптимизация / Москва – Изд-во «Мир», 1985. – 509 с.
5. Данциг Дж. Линейное программирование его применения и обобщения / Дж. Данциг. – М.: Изд-во «Прогресс», 1966. – 600 с.
6. Юдин Д.Б., Гольдштейн Е.Г. Задачи и методы линейного программирования / Москва – Изд-во «Советское радио», 1961. – 491 с.
7. Зуховицкий С.И., Авдеева Л.И. Линейное и выпуклое программирование / Москва – Изд-во «Наука», 1967. – 460 с.
8. Гантмахер Ф.Р. Теория матриц. – 5-е изд. – М.: Физматлит, 2004. – 559 с.
9. Вылегжанин О.Н. Сравнительный анализ возможностей линейного и квадратичного программирования в количественной спектрофотометрии смесей неполностью известного качественного состава // Журнал прикладной спектроскопии. – 1990. – Т. 52. – № 6. – С. 984–988.
10. Небаба С.Г., Вылегжанин О.Н. Построение и исследование алгоритма поиска первой крайней подсистемы для заданной совместной системы линейных неравенств // Известия Томского политехнического университета. – 2012. – Т. 320. – № 5. – С. 17–21.
11. Цыганков А.А. Новые условия экстремума для гладких задач с ограничениями в форме равенств // Журнал вычислительной математики и математической физики. – 2001. – Т.41. – № 10. – С. 1474–1481.

12. Альберт А. Регрессия, псевдоинверсия и рекуррентное оценивание // Москва – Изд-во «Наука», 1977. – 225 с.
13. Васильев Ф.П., Иваницкий А.Ю. Линейное программирование. – М.: Факториал, 1998. – 323 с.5
14. Черников С. Н. Линейные неравенства. // Москва – Изд-во «Наука», 1968. – 488 с.
15. СН 2.2.4/2.1.8.562–96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории застройки.
16. СанПиН 2.2.548-96. Гигиенические требования к микроклимату производственных помещений.
17. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.
18. Гигиенические требования к видеодисплейным терминалам, персональным электронно-вычислительным машинам и организации работы. Выписки из СаНпиН 2.2.2.542-96.
19. СНиП 23-05-95 Естественное и искусственное освещение.
20. ГОСТ 12.1.019 (с изм. №1) ССБТ. Электробезопасность. Общие требования и номенклатура видов защиты.
21. Федеральный закон Российской Федерации от 22 июля 2008 г. N 123-ФЗ "Технический регламент о требованиях пожарной безопасности".
22. Федеральный классификационный каталог отходов [Электронный ресурс] – 2013. – Режим доступа: http://www.ecoguild.ru/, свободный. – Загл. с экрана.
23. Об утверждении правил обращения с отходами производства и потребления в части осветительных устройств, электрических ламп, ненадлежащие сбор, накопление, использование, обезвреживание, транспортирование и размещение которых может повлечь причинение вреда жизни, здоровью граждан, вреда животным, растениям и окружающей среде: Постановление Правительства Российской Федерации от 3 сентября 2010 года № 681.
24. ТОИ Р-45-084-01. Типовая инструкция по охране труда при работе на персональном компьютере.

ПРИЛОЖЕНИЯ

Приложение А – Процедура перемножения матриц

```
void multplication_matrix (double **A, int ma, int na, double **B, int mb, int nb, double
**C)
{
	for (int i = 0; i < ma; i++) {
		for (int j = 0; j < nb; j++) {
			C[i][j] = 0;
			for (int j1 = 0; j1 < mb; j1++) {
				C[i][j] = C[i][j] + (A[i][j1] * B[j1][j]);}}}
```

Приложение Б – Процедура разделения матрицы

```
void column_swaps (double **A, double **A2, int ma, int na, double **An, double **A2n, int
ma2, int z, int k, double **P, double **Pn) {

	//Перенос столбца с максимальной нормой из A1 в An
	for(int i=0; i<ma; ++i){
			An[i][z-1]=A[i][k];}
	//Перенос столбца с максимальной нормой из A2 в An
	for(int i=0; i<ma2; ++i){
			A2n[i][z-1]=A2[i][k];}
	//Удаление столбца с максимальной нормой из A1
	for(int j=k; j<na; ++j){
		for(int i=0; i<ma; ++i){
			if (j == na-1) {
				A[i][j] = 0;}
			else
			A[i][j] = A[i][j+1];}

	//Удаление столбца с номером столбца с максимальной нормой из A2
		for(int i=0; i<ma2; ++i){
			if (j == na-1) {
				A2[i][j] = 0;}
			else
			A2[i][j] = A2[i][j+1];}}

	//Перенос элемента с номером столбца с максимальной нормой из P в Pn
	Pn[z-1][0]=P[k][0];
	//Удаление элемента с номером столбца с максимальной нормой из P
	for(int i=k; i<na; ++i){
		if (i == na-1) {
			P[i][0] = 0;}
		else
			P[i][0] = P[i+1][0];}

}
```

Приложение В – Процедура поиска обратной матрицы

```
void inversion(double **A, double **Ainv, int N){
    double temp;
    double **E = new double *[N];

    for (int i = 0; i < N; i++)
        E[i] = new double [N];
    for (int i=0; i<N; i++)
        for (int j=0; j<N; j++){
            E[i][j] = 0.0;
            if (i == j)
                E[i][j] = 1.0;}
    for (int k = 0; k < N; k++){
        temp = A[k][k];
        for (int j=0; j<N; j++){
            A[k][j]/= temp;
            E[k][j]/= temp;}
        for (int i=k+1; i<N; i++){
            temp = A[i][k];
            for (int j =0; j<N; j++){
                A[i][j]-= A[k][j]*temp;
                E[i][j]-= E[k][j]*temp;}}}
    for (int k=N-1; k>0; k--){
        for (int i = k-1; i>=0; i--){
            temp=A[i][k];
            for (int j=0; j<N; j++){
                A[i][j]-=A[k][j]*temp;
                E[i][j]-=E[k][j]*temp;}}}
    for (int i=0; i<N; i++)
        for (int j=0; j<N; j++)
            Ainv[i][j]=E[i][j];
    for (int i=0; i<N; i++)
        delete[]E[i];
    delete[]E;
}
```

Приложение Г – Процедура поиска столбца с максимальной нормой

```
void find_max_norm (double **A, int ma, int na, int &k, double &norm){

	double **a = new double*[ma];
	for(int i=0; i<ma; ++i){
		a[i] = new double[1];}
	double **aT = new double*[1];
	for(int i=0; i<1; ++i){
		aT[i] = new double[ma];}
	double **aP = new double*[1];
	for(int i=0; i<1; ++i){
		aP[i] = new double[1];}

	double summ = 0;
	norm = 0;
	cout<<endl;
	for(int j=0; j<na; ++j){
		summ = 0;
		for(int i=0; i<ma; ++i){
			a[i][0] = A[i][j];
			aT[0][i] = a[i][0];}
			multplication_matrix(aT, 1, ma, a, ma, 1, aP);
			summ = aP[0][0];
			if (norm < (sqrt(summ))) {
				norm = sqrt(summ);
					k = j;}
		cout<<endl<<"current_norm = "<<sqrt(summ)<<endl<<"max_norm = "<<norm;}
}
```

Приложение Д – Процедура вывода матрицы на экран

```
void print_matrix (double **A, int ma, int na, string text) {
       cout<<endl<<endl<<text;
       for(int i=0; i<ma; ++i){
                     for(int j=0; j<na; ++j){
                     if (j==0) cout<<endl;
                     cout<<A[i][j]<<" ";}}
```

Приложение Е – Процедура поиска псевдообратной матрицы

```
void pseudoinversion (double **A, int m, int n, double **APlus) {

	int c=0;

	double **U = new double*[m];
	for(int i=0; i<m; ++i){
		U[i] = new double[n];}
	double **W = new double*[n];
	for(int i=0; i<n; ++i){
		W[i] = new double[n];}
	double **V = new double*[n];
	for(int i=0; i<n; ++i){
		V[i] = new double[n];}
	double **WV = new double*[n];
	for(int i=0; i<n; ++i){
		WV[i] = new double[n];}
	double **UWV = new double*[m];
	for(int i=0; i<m; ++i){
		UWV[i] = new double[n];}
	double **Winv = new double*[n];
	for(int i=0; i<n; ++i){
		Winv[i] = new double[n];}


	MatDoub A2 (m, n);

	for(int i=0; i<m; ++i){
		for(int j=0; j<n; ++j){
			A2[i][j]=A[i][j];}}

	print_matrix(A, m, n, "A");

	cout<<endl<<"A2"<<endl;
	for(int i=0; i<m; ++i){
		for(int j=0; j<n; ++j){
			if (j==0) cout<<endl;
			cout<<A2[i][j]<<" ";}}

	SVD_Alt MakeSVD (A2);

	for(int i=0; i<m; ++i){
		for(int j=0; j<n; ++j){
			U[i][j]=MakeSVD.u[i][j];}}

	print_matrix(U, m, n, "U");

	for(int i=0; i<n; ++i){
		for(int j=0; j<n; ++j){
			if (i==j){
				if (abs(MakeSVD.w[i])>0.0001){
				c+=1;}
				W[i][j]= MakeSVD.w[i];}
			else {
				W[i][j]=0;}}}

	print_matrix(W, n, n, "W");

	for(int i=0; i<n; ++i){
		for(int j=0; j<n; ++j){
			V[j][i]=MakeSVD.v[i][j];}}

	print_matrix(V, n, n, "V");
```

```
	inversion (W, Winv, c);

	for(int i=0; i<c; ++i){
		for(int j=0; j<c; ++j){
			W[i][j]=Winv[i][j];}}

	print_matrix(Winv, n, n, "Winv");

	multplication_matrix(W, n, n, V, n, n, WV);
	multplication_matrix(U, m, n, WV, n, n, UWV);

	print_matrix(UWV, m, n, "UWV_A");

	multplication_matrix(W, n, n, V, n, n, WV);
	multplication_matrix(U, m, n, WV, n, n, UWV);

	print_matrix(UWV, m, n, "UWV");

	for(int i=0; i<m; ++i){
		for(int j=0; j<n; ++j){
			APlus[j][i]=UWV[i][j];}}
}
}
```

Приложение Ж – Процедура преобразования квадратичного и линейного функционала

```
void functional_conversion_P2_and_Q (double **W, int na, int z, double **Qluc, double
**Qruc, double **Qldc, double **Qrdc, double **Q, double **PfT, double **B) {
	double **WT = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		WT[i] = new double[z];}
	double **BT = new double*[1];
	for(int i=0; i<1; ++i){
		BT[i] = new double[z];}
	double **QlucT = new double*[z];
	for(int i=0; i<z; ++i){
		QlucT[i] = new double[z];}
	double **QldcT = new double*[z];
	for(int i=0; i<z; ++i){
		QldcT[i] = new double[na-z];}
	double **QlucW = new double*[z];
	for(int i=0; i<z; ++i){
		QlucW[i] = new double[na-z];}
	double **WTQlucW = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		WTQlucW[i] = new double[na-z];}
	double **QldcW = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		QldcW[i] = new double[na-z];}
	double **WTQruc = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		WTQruc[i] = new double[na-z];}
	double **QlucQlucTSum = new double*[z];
	for(int i=0; i<z; ++i){
		QlucQlucTSum[i] = new double[z];}
	double **QlucQlucTSumW = new double*[z];
	for(int i=0; i<z; ++i){
		QlucQlucTSumW[i] = new double[na-z];}
	double **PfTdB = new double*[z]; //PfT devided by B
	for(int i=0; i<z; ++i){
		PfTdB[i] = new double[na-z];}

//транспонирования для следующих действий
	for(int i=0; i<z; ++i){
			for(int j=0; j<na-z; ++j){
				WT[j][i]=W[i][j];}}
	for(int i=0; i<z; ++i){
			for(int j=0; j<z; ++j){
				QlucT[j][i]=Qluc[i][j];}}
	for(int i=0; i<na-z; ++i){
			for(int j=0; j<z; ++j){
				QldcT[j][i]=Qldc[i][j];}}
	for(int i=0; i<z; ++i){
				BT[i][0]=B[0][i];}

//Q-функционал
	multplication_matrix(Qluc, z, z, W, z, na-z, QlucW);
	multplication_matrix(WT, na-z, z, QlucW, z, na-z, WTQlucW);
	multplication_matrix(Qldc, na-z, z, W, z, na-z, QldcW);
	multplication_matrix(WT, na-z, z, Qruc, z, na-z, WTQruc);
	for(int i=0; i<na-z; ++i){
			for(int j=0; j<na-z; ++j){
				Q[i][j] = WTQlucW[i][j] + QldcW[i][j] + WTQruc[i][j] +
Qrdc[i][j];}}

//P2-функционал
```

```
	for(int i=0; i<z; ++i){
			for(int j=0; j<z; ++j){
				QlucQlucTSum[i][j]=Qluc[i][j]+QlucT[i][j];}}

	multplication_matrix(QlucQlucTSum, z, z, W, z, na-z, QlucQlucTSumW);

	for(int i=0; i<z; ++i){
			for(int j=0; j<na-z; ++j){
				PfTdB[i][j] = QldcT[i][j] + Qruc[i][j] - QlucQlucTSumW[i][j];}}

	multplication_matrix(BT, 1, z, PfTdB, z, na-z, PfT);
}
```

Приложение 3 – Процедура поиска проекции начальной точки на грань области допустимых значений

```
void X1_Calc (double **X, int na, double **B2, double **A2, int ma2, int z) {

	double up, all;

	double **a2 = new double*[1];
		for(int i=0; i<1; ++i){
			a2[i] = new double[na-z];}
	double **a2Tall = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			a2Tall[i] = new double[1];}
	double **a2T = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			a2T[i] = new double[1];}
	double **a2TX = new double*[1];
		for(int i=0; i<1; ++i){
			a2TX[i] = new double[1];}
	double **a2a2T = new double*[1];
		for(int i=0; i<1; ++i){
			a2a2T[i] = new double[1];}

	for(int i=0; i<ma2; ++i){
		for(int j=0; j<na-z; ++j){
			a2[0][j]=A2[i][j];
			a2T[j][0]=A2[i][j];}
	multplication_matrix(a2, 1, na-z, X, na-z, 1, a2TX);
	up = B2[i][0]-a2TX[0][0];
	multplication_matrix(a2, 1, na-z, a2T, na-z, 1, a2a2T);
	all=up/a2a2T[0][0];
	for(int i=0; i<na-z; ++i){
		a2Tall[i][0]=all*a2T[i][0];}
	for(int i=0; i<na-z; ++i){
		X[i][0]=X[i][0]+a2Tall[i][0];}}
}
```

Приложение И – Процедура поиска ближайшей к начальной точке вершины

```
void X1_vertex (double **A2, double **X, double **B2, int ma2, int na, int z) {

	double up, all; bool y=false; int c = 0;

	double **a2 = new double*[1];
		for(int i=0; i<1; ++i){
			a2[i] = new double[na-z];}
	double **Ra = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			Ra[i] = new double[na-z];}
	double **A2Plus = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			A2Plus[i] = new double[ma2];}

	double **a2TRa = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			a2TRa[i] = new double[1];}
	double **a2T = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			a2T[i] = new double[1];}
	double **a2TX = new double*[1];
		for(int i=0; i<1; ++i){
			a2TX[i] = new double[1];}
	double **a2Ra = new double*[1];
		for(int i=0; i<1; ++i){
			a2Ra[i] = new double[1];}
	double **A2a2T = new double*[ma2];
		for(int i=0; i<ma2; ++i){
			A2a2T[i] = new double[1];}
	double **A2PlusA2a2T = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			A2PlusA2a2T[i] = new double[1];}

	do {
	for(int i=0; i<ma2; ++i){
		for(int j=0; j<na-z; ++j){
			a2[0][j]=A2[i][j];
			a2T[j][0]=A2[i][j];}

	pseudoinversion (A2, ma2, na-z, A2Plus);
	print_matrix(A2Plus,na-z, ma2, "A2Plus");


	multplication_matrix(A2, ma2, na-z, a2T, na-z, 1, A2a2T);
	print_matrix(A2a2T,ma2, 1, "A2a2T");
	multplication_matrix(A2Plus, na-z, ma2, A2a2T, ma2, 1, A2PlusA2a2T);
	print_matrix(A2PlusA2a2T,na-z,1, "a2PlusA2a2T");

	for(int i=0; i<na-z; ++i){
		Ra[i][0] = a2T[i][0] - A2PlusA2a2T[i][0];}
	print_matrix(Ra, na-z, na-z, "Ra");

	multplication_matrix(a2, 1, na-z, X, na-z, 1, a2TX);
	print_matrix(a2TX,1,1, "a2TX");
	up = B2[i][0]-a2TX[0][0];
	multplication_matrix(a2, 1, na-z, Ra, na-z, 1, a2Ra);
	print_matrix(a2Ra,1,1, "a2Ra");
	all=up/a2Ra[0][0];
	for(int i=0; i<na-z; ++i){
		a2TRa[i][0]=all*Ra[i][0];}
```

```
        print_matrix(a2TRa,na-z,1, "a2TRa");
        for(int i=0; i<na-z; ++i){
               X[i][0]=X[i][0]+a2TRa[i][0];}}

        print_matrix(X,na-z,1, "X");

        basics(A2, X, B2, y, ma2, na-z);
        cout<<endl<<y<<endl;
        //c=c+1;
        } while (y==false);

}
```

Приложение К – Процедура поиска условного экстремума методом оператора проектора

```
void conditional_extremum (double **A2, double **Q, int ma2, int na, int z, double **P2,
double **X, double **B2){

	bool check_U = false, check_I; int k = 0;

	double **A2bPlus = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			A2bPlus[i] = new double[ma2];}
	double **Y = new double*[ma2];
		for(int i=0; i<ma2; ++i){
			Y[i] = new double[1];}
	double **A2b = new double*[ma2];
		for(int i=0; i<ma2; ++i){
			Y[i] = new double[na-z];}
	double **U = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			U[i] = new double[1];}

	double **A2bPlusT = new double*[ma2];
		for(int i=0; i<ma2; ++i){
			A2bPlusT[i] = new double[na-z];}
	double **QX = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			QX[i] = new double[1];}
	double **G = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			G[i] = new double[1];}
	double **Alpha = new double*[1];
		for(int i=0; i<1; ++i){
			Alpha[i] = new double[1];}
	double **S = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			S[i] = new double[1];}
	double **A2PlusA2 = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			A2PlusA2[i] = new double[1];}
	double **R = new double*[na-z];
		for(int i=0; i<na-z; ++i){
			R[i] = new double[na-z];}

	double **a2bT = new double*[1];
		for(int i=0; i<1; ++i){
			a2bT[i] = new double[na-z];}
	double **a2bTX = new double*[1];
		for(int i=0; i<1; ++i){
			a2bTX[i] = new double[1];}
	double **a2bTS = new double*[1];
		for(int i=0; i<1; ++i){
			a2bTS[i] = new double[1];}

	multplication_matrix(A2, ma2, na-z, X, na-z, 1, Y);

	for(int i=0; i<ma2; ++i){
		if(Y[i][0]==B2[i][0]){
			for(int j=0; j<na-z; ++j){
				A2b[k][j]=A2[i][j];}
			k+=1;}}

	pseudoinversion (A2b, k, na-z, A2bPlus);
```

```
        for(int i=0; i<na-z; ++i){
               for(int j=0; j<k; ++j){
                      A2bPlusT[j][i]=A2bPlus[i][j];}}

        multplication_matrix(Q, na-z, na-z, X, na-z, 1, QX);

        for(int i=0; i<na-z; ++i){
               G[i][0]=2*QX[i][0]+P2[i][0];}

        multplication_matrix(A2bPlusT, k, na-z, X, na-z, 1, U);

         do{

               check_U=true;

               multplication_matrix(A2bPlus, na-z, k, A2b, k, na-z, R);

               multplication_matrix(Q, na-z, na-z, X, na-z, 1, QX);
               for(int i=0; i<na-z; ++i){
                      G[i][0]=2*QX[i][0]+P2[i][0];}
               multplication_matrix(R, na-z, na-z, G, na-z, 1, S);

               multplication_matrix(a2bT, 1, na-z, X, na-z, 1, a2bTX);
               multplication_matrix(a2bT, 1, na-z, S, na-z, 1, a2bTS);
               Alpha[0][0]=(-a2bTX[0][0]/a2bTS[0][0]);

               for(int i=0; i<na-z; ++i){
                      X[i][0]=X[i][0]+(Alpha[0][0]*S[i][0]);}

               multplication_matrix(A2bPlusT, k, na-z, X, na-z, 1, U);

               for(int i=0; i<k; ++i){
                      if(U[i][0]<0){
                             check_U=false;}}
         } while (check_U == false);
}
```

```
int _tmain(int argc, _TCHAR* argv[])
{
	int ma; int ma2; int na; int z = 0; int k = 0; double norm; double tol; bool check = true; //(z - итерация, k - номер выбранного столбца из A, k_C - номер выбранного столбца из C)

	//Ввод точности
	cout<<"tol = ";
	cin>>tol;
	//Ввод размерностей матриц
	cout<<"ma = ";
	cin>>ma;
	cout<<"ma2 = ";
	cin>>ma2;
	cout<<"na = ";
	cin>>na;
	//~

	//Выделение памяти

	//Ограничения
	//Матрица ограничений-равенств A1
	double **A = new double*[ma];
	for(int i=0; i<ma; ++i){
		A[i] = new double[na];}
	//Вектор свободных элементов B1
	double **B = new double*[ma];
	for(int i=0; i<ma; ++i){
		B[i] = new double[1];}

	//Матрица ограничений-неравенств A2
	double **A2 = new double*[ma2];
	for(int i=0; i<ma2; ++i){
		A2[i] = new double[na];}
	//Вектор свободных элементов B2
	double **B2 = new double*[ma2];
	for(int i=0; i<ma2; ++i){
		B2[i] = new double[1];}
	//~

	//Целевой функционал
	//Матрица коэффициентов квадратичной части Q
	double **Q = new double*[na];
	for(int i=0; i<na; ++i){
		Q[i] = new double[na];}
	//Вектор коэффициентов линейной части P
	double **P = new double*[na];
	for(int i=0; i<na; ++i){
		P[i] = new double[1];}

	//Рабочая точка
	double **X = new double*[na];
	for(int i=0; i<na; ++i){
		X[i] = new double[1];}
	//~

	//Вспомогательные матрицы
	//Матрица независимых векторов A1
	double **An = new double*[ma];
	for(int i=0; i<ma; ++i){
		An[i] = new double[na];}
	//Матрица независимых векторов A2
```

```
double **A2n = new double*[ma2];
for(int i=0; i<ma2; ++i){
	A2n[i] = new double[na];}
//Псевдообратная матрица к An
double **AnPlus = new double*[na];
for(int i=0; i<na; ++i){
	AnPlus[i] = new double[ma];}

double **C = new double*[ma];
for(int i=0; i<ma; ++i){
	C[i] = new double[na];}
//Псевдообратная матрица к C
double **CPlus = new double*[1];
for(int i=0; i<1; ++i){
	CPlus[i] = new double[ma];}

//Вектор независимых элементов P
double **Pn = new double*[na];
for(int i=0; i<na; ++i){
	Pn[i] = new double[1];}
//Первая часть линейного функционала
double **P1 = new double*[1];
for(int i=0; i<1; ++i){
	P1[i] = new double[na];}
//Вторая часть линейного функционала
double **P2 = new double*[1];
for(int i=0; i<1; ++i){
	P2[i] = new double[na];}
//~

//Ввод данных
//Квадратичная часть функционала Q
for(int i=0; i<na; ++i){
	for(int j=0; j<na; ++j){
		cout<<"Q["<<i+1<<","<<j+1<<"] = ";
		cin>>Q[i][j];}}
//Линейная часть функционала P
for(int i=0; i<na; ++i){
	for(int j=0; j<1; ++j){
		cout<<"P["<<i+1<<","<<j+1<<"] = ";
		cin>>P[i][j];}}
//Матрица ограничений-равенств A1
for(int i=0; i<ma; ++i){
	for(int j=0; j<na; ++j){
		cout<<"A1["<<i+1<<","<<j+1<<"] = ";
		cin>>A[i][j];}}
//Матрица ограничений-неравенств A2
for(int i=0; i<ma2; ++i){
	for(int j=0; j<na; ++j){
		cout<<"A2["<<i+1<<","<<j+1<<"] = ";
		cin>>A2[i][j];}}
//Вектор свободных элементов B1
for(int i=0; i<ma; ++i){
	for(int j=0; j<1; ++j){
		cout<<"B1["<<i+1<<","<<j+1<<"] = ";
		cin>>B[i][j];}}
//Вектор свободных элементов B2
for(int i=0; i<ma2; ++i){
	for(int j=0; j<1; ++j){
		cout<<"B2["<<i+1<<","<<j+1<<"] = ";
		cin>>B2[i][j];}}
//Начальная точка Xo
for(int i=0; i<na; ++i){
	for(int j=0; j<1; ++j){
```

```
            cout<<"Xo["<<i+1<<","<<j+1<<"] = ";
            cin>>X[i][j];}}
//~

//Выводы на экран начальных данных
//Вывод начальной матрицы ограничений-равенств A1
print_matrix (A, ma, na, "A1");
//Вывод начальной матрицы ограничений-неравенств A2
print_matrix (A2, ma2, na, "A2");

//Вывод вектора свободных элементов B1
print_matrix (B, ma, 1, "B1");
//Вывод вектора свободных элементов B2
print_matrix (B2, ma2, 1, "B2");

//Вывод квадратичной части функционала
print_matrix (Q, na, na, "Q");
//Вывод линейного функционала
print_matrix (P, na, 1, "P");
//Вывод Xo на экран
print_matrix (X, na, 1, "Xo");
//~

//Тело программы
//Функция поиска столбца с максимальной нормой
find_max_norm (A, ma, na, k, norm);
//Номер итерации
z = 1;

//Вывод номера столбца с максимальной нормой из A1
cout<<endl<<"k = "<<k;

column_swaps (A, A2, ma, na, An, A2n, ma2, z, k, P, Pn);

//----------------------------------------

//Вывод A1n на экран
print_matrix (An, ma, z, "A1n");
//Вывод A2n на экран
print_matrix (A2n, ma2, z, "A2n");
//Вывод A1 на экран
print_matrix (A, ma, na, "A1_sub");
//Вывод A2 на экран
print_matrix (A2, ma2, na, "A2_sub");
//Вывод Pn на экран
print_matrix (Pn, z, 1, "Pn");
//Вывод P на экран
print_matrix (P, na, 1, "P_sub");

//----------------------------------------

//Поиск псевдообратной матрицы An
find_pseudo_inverse (An, AnPlus, 0, ma);

//Вывод псевдообратной матрицы к An
print_matrix (AnPlus, z, ma, "An+");

norm = 1000;

//Рекуррентный процесс
do {
      if (na > z) {
//Расчёт матрицы C
C_calc (A, ma, na, z, An, AnPlus, C);
```

```
        //Вывод матрицы C на экран
        print_matrix (C, ma, na-z, "C");

        //Поиск номера столбца с максимальной нормой
        find_max_norm (C, ma, na-z, k, norm);

        //Номер столбца с максимальной нормой из C
        (?)
//      k_C = k;

        cout<<endl<<"k = "<<k+1;
        cout<<endl<<"z = "<<z;

        if ((norm > tol)) {
        //Итерация +1
               z=z+1;

        //Перестановка столбцов в матрицах A, An, P, Pn, A2 и A2n
        column_swaps (A, A2, ma, na, An, A2n, ma2, z, k, P, Pn);

        //Поиск псевдообратной матрицы к C
        find_pseudo_inverse (C, CPlus, k, ma);

        //Вывод псевдообратной матрицы к C
        print_matrix (CPlus, 1, ma, "C+");

        //Пересчёт псевдообратной матрицы An+ с учётом C+
        refind_pseudoinversion_An (A, ma, na, z, CPlus, AnPlus, k);

        //Добавление к An+ C+
        for(int i=0; i<ma; ++i){
                     AnPlus[z-1][i]=CPlus[0][i];}

        //Вывод An+ на экран
        print_matrix (AnPlus, z, ma, "An+");

        //-------------------------------------------------------------

        //Вывод An на экран
        print_matrix (An, ma, z, "A1n");
        //Вывод A2n на экран
        print_matrix (A2n, ma2, z, "A2n");
        //Вывод A1 на экран
        print_matrix (A, ma, na, "A1_sub");
        //Вывод A2 на экран
        print_matrix (A2, ma2, na, "A2_sub");
        //Вывод Pn на экран
        print_matrix (Pn, z, 1, "Pn");
        //Вывод P на экран
        print_matrix (P, na, 1, "P_sub");

        //-------------------------------------------------------------

        //Проверка условия, что матрица ограничений-равенств A1 не пуста
        //~
        }

        check = true;
        for(int i=0; i<ma; ++i){
               for(int j=0; j<na; ++j){
                      if (A[i][j] != 0) {
                             check = false;}}}
```

```
	print_matrix (An, ma, z, "A1n_end");

	print_matrix (A2n, ma2, z, "A2n_end");

	print_matrix (A, ma, na, "A1_end");

	print_matrix (A2, ma2, na, "A2_end");

		}
		if (check == true)
		break;
	} while (norm > tol);
	//Конец рекуррентного процесса

	double **B1new = new double*[z];
	for(int i=0; i<z; ++i){
		B1new[i] = new double[1];}

	//Пересчёт столбца свободных элементов B1
	multplication_matrix(AnPlus, z, ma, B, ma, 1, B1new);

	//Вывод пересчитанного B1
	print_matrix (B1new, z, 1, "B_new");

	//Проверка условия, что матрица ограничений-равенств A1 не пуста
	if (check == true) {
		cout<<endl<<"The solution is found, matrix A is empty"<<endl;

		X1_Calc (X, na, B2, A2, ma2, 0);
		print_matrix (X, na, 1, "X1");
	}
	else {

	double **W = new double*[z];
	for(int i=0; i<z; ++i){
		W[i] = new double[na-z];}

	//Расчёт первой части линейного функционала
	functional_conversion_P1 (A, ma, na, z, Pn, P, AnPlus, P1);

	print_matrix (P1, 1, na-z, "P1");

	//Разбиения функционала Q на 4 части: luc  - left up corner, ruc - right up corner,
ldc  - left down corner, rdc - right down corner
	//-----------------------------
	double **Qluc = new double*[z];
	for(int i=0; i<z; ++i){
		Qluc[i] = new double[z];}

	for(int i=0; i<z; ++i){
			for(int j=0; j<z; ++j){
				Qluc[i][j] = Q[i][j];}}

		double **Qruc = new double*[z];
	for(int i=0; i<z; ++i){
		Qruc[i] = new double[na-z];}

	for(int i=0; i<na-z; ++i){
			for(int j=0; j<na-z; ++j){
				Qruc[i][j] = Q[i][j];}}

		double **Qldc = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		Qldc[i] = new double[z];}
```

```
	for(int i=0; i<na-z; ++i){
			for(int j=0; j<z; ++j){
				Qldc[i][j] = Q[i][j];}}

		double **Qrdc = new double*[na-z];
	for(int i=0; i<na-z; ++i){
		Qrdc[i] = new double[na-z];}

	for(int i=0; i<na-z; ++i){
			for(int j=0; j<na-z; ++j){
				Qrdc[i][j] = Q[i][j];}}
	//-----------------------------------

	//Расчёт матрицы W
	multplication_matrix(AnPlus, z, ma, A, ma, na-z, W);

	//Расчёт второй части линейного функционала и пересчёт квадратичной части функционала
	functional_conversion_P2_and_Q(W, na, z, Qluc, Qruc, Qldc, Qrdc, Q, P2, B1new);

	print_matrix (P2, 1, na-z, "P2");

	//Пересчёт линейной части функционала
	for(int j=0; j<na-z; ++j){
		P[j][0] = P1[0][j] + P2[0][j];}

	//Вывод на экран пересчитанной линейной части
	print_matrix (P, na-z, 1, "Pnew");

	print_matrix (Q, na-z, na-z, "Qnew");

	X1_Calc (X, na, B2, A2, ma2, z);
	X1_vertex (A2, X, B2, ma2, na, z);

	print_matrix (X, na-z, 1, "X1");

//	conditional_extremum (A2, Q, ma2, na, z, P2, X, B2);

	}
	cout<<endl;
	system ("Pause");
	return 0;
}
```