СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ПРОЦЕССЕ ОБУЧЕНИЯ

(коллективная монография)

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ПРОЦЕССЕ ОБУЧЕНИЯ

Коллективная монография

Под редакцией
доктора физико-математических наук, профессора,
директора лаборатории системного анализа
Миронова Валентина Васильевича

Рязань 2017

УДК 378.4:004

Авторы:

В. В. Миронов
В.В. Гришина
А.И. Заволокин
А.К. Розанов
С.А. Нелюхин

Под редакцией
доктора физико-математических наук, профессора,
директора лаборатории системного анализа
Миронова Валентина Васильевича

Рецензенты:
Куракин Дмитрий Владимирович – д-р техн. наук, профессор, лауреат премии Правительства РФ в области образования, почетный работник науки и техники РФ, вице-президент Академии информатизации образования, советник директора ФГАУ ГНИИ ИТТ «Информика», главный редактор журнала «Информатизация образования и науки», г. Москва.

Мухин Александр Владимирович, канд. физ.-мат. наук, ведущий научный сотрудник Федерального исследовательского центра «Информатика и управление» РАН, г. Москва.

Современные технологии в процессе обучения: монография/ под ред. проф. В. В. Миронова. – Рязань, Book Jet, 2017. – 160 с.

В монографии раскрыты аспекты практического использования современных компьютерных технологий в процессе обучения на основе общих закономерностей и анализа образовательной деятельности Рязанского государственного радиотехнического университета.

Для преподавателей и аспирантов технических и педагогических вузов.

Печатается по решению Рязанского физико-математического общества.

ISBN _____

Оглавление

Введение

Вряд ли кто из квалифицированных специалистов или «простых» пользователей информационных ресурсов станет сегодня отрицать, что процесс информатизации всех сторон жизни человека и общества в целом принимает всеобщий, лавинообразный, а зачастую и неуправляемый характер.

Более того, мир стоит на пороге новой технической революции, последствия которой для человека трудно адекватно оценить, предсказать. Речь может идти даже о создании (именно создании) нового человека.

Отнюдь не случайно ведущие ученые мира (а за ними и деловые люди) призвали уделить предельное внимание безопасности и разработке этических норм ввиду реальности появления сверхкомпьютеров и, как следствие, созданию искусственного интеллекта. Угроза реальна и велика.

Инициированное американским физиком-теоретиком С. Хокингом открытое письмо к мировой общественности по всей этой проблематике в свое время подписали представители современной науки из лучших университетов мира: Массачусетского, Гарвардского, Стэнфордского, Кембриджского, Оксфордского, а также главы проектов Skype (*прим.* Skype был куплен Microsoft в 2011 г.), Space X, позднее к обращению присоединились главы всемирно известных компаний Google, Microsoft и IBM.

Россия, как всегда, не торопится с промышленными (в отличие от социальных) революциями. Вот, к примеру, TOP 10 ведущих компаний мира по проблеме искусственного интеллекта: Numenta, Scaled Inference, Amazon, Nuance, Henson Robotics, Microsoft, Facebook, Apple, Google, IBM. Как видим, российских представителей здесь нет.

Обратимся ввиду названной общей проблематики к частной проблеме тотальной информатизации всего образования. С начала 2000-х годов уже повсеместны применение и внедрение компьютеров в жизнь общества выдвинули информацию и знание вообще на передний план социального и экономического развития. Информация в нем является главным экономическим ресурсом, а информационный сектор выходит на первое место по темпам развития, по числу занятых специалистов, по доле капиталовложений.

Эти процессы предполагают наличие развитой инфраструктуры, обеспечивающей создание достаточных информационных ресурсов, в первую очередь в науке и системе образования. В связи с этим в образовательной сфере деятельности человека предпринимаются настойчивые попытки внедрить компьютеризированные технологии и в такую тонкую «материю», как процесс общения учителя и его ученика. Проблема, на наш взгляд, состоит в том, что этими новыми информационными технологиями хотят ограничить или сопроводить весь процесс обучения, который, вне всякого сомнения, неотделим от педагогики!

Нет ли здесь, во всем этом процессе, серьезной ошибки, столь присущей русскому сознанию, смело и безоглядно бросающемуся от одного «устаревшего» к другому «новому»?

Благодаря внедрению в образовательные учреждения современных средств информационных и телекоммуникационных технологий и использованию их в качестве нового педагогического инструмента, позволяющего увеличить эффективность образовательного процесса, педагоги получили новые средства и организационные формы учебной работы, которые в дальнейшем стали использоваться повсеместно и способны поддерживать практически все стадии образовательного процесса.

Но эти технологии служат только помощниками и никогда не смогут заменить учителей, хотя такие тенденции не просто существуют, а стали доминирующими, профессионального учителя хотят заменить (а где-то уже заменяют) инженером.

В данной работе как раз и предпринят анализ проблемы информатизации образования и угроз, которые она (информатизация) способна нести в себе. Рассмотрены принципы и границы информатизации и тех основ образования и воспитания, что стоят за информатизацией, а за ней стоит человек, учитель.

Работы отечественных и зарубежных авторов по вопросам информатизации образования свидетельствуют о возможности повышения его эффективности. Новые результаты в области информатизации образования, применения техники в педагогической деятельности получены в России Я.А. Вагрaменко, И.В. Востроknутовым, С.Г. Григорьевым, В.В. Гриншкуном, В.С. Гуровым, А.П. Ершовым, О.А. Козловым, Е.С. Подписчиковым и другими учеными, а среди зарубежных специалистов – Р. Вильямом, Н. Виртом, Д. Гриссом, Э. Дейкстра, П. Деннингом, Б. Хатнером и другими.

Авторы выражают надежду, что представленные в монографии результаты исследования будут интересны и полезны преподавателям университетов, которые уже используют средства информатизации в профессиональной преподавательской деятельности.

Материал по главам (и авторам) распределен следующим образом.

Глава 1. В.В. Миронов, В.В. Гришина
ИНФОРМАТИЗАЦИЯ ОБРАЗОВАНИЯ: ДОСТИЖЕНИЯ И ПРОБЛЕМЫ

Аннотация. Глава посвящена философскому осмыслению проблемы внедрения информационных технологий в процесс обучения. Рассмотрены различные направления, достижения, проблемы и постулаты современного образования, в том числе и не подлежащие компьютеризации.
Ключевые слова: образование, информатизация, компьютеризация.

Глава 2. А.И. Заволокин, В.В Миронов, А.К. Розанов
ВОПРОСЫ ИНФОРМАТИЗАЦИИ ОБРАЗОВАНИЯ В ПРОЦЕССЕ ОБУЧЕНИЯ СТУДЕНТОВ РУССКО-АНГЛИЙСКОМУ И АНГЛО-РУССКОМУ ПЕРЕВОДУ С АВТОМАТИЗИРОВАННЫМ КОНТРОЛЕМ ЗНАНИЙ ЧЕРЕЗ ПРИМЕНЕНИЕ КОМПЬЮТЕРНЫХ ПРОГРАММ, РЕАЛИЗУЮЩИХ АЛГОРИТМЫ АКТИВНОЙ И ПАССИВНОЙ ГРАММАТИК

Аннотация. Глава посвящена вопросам происхождения естественного языка, выведению грамматических категорий и новым методам обучения русско-английскому и англо-русскому переводу с использованием математических принципов алгоритмизации (перевода). Дается теоретическое обоснование активной и пассивной грамматик, показаны возможные варианты их практического использования в учебном процессе, в том числе с применением компьютерных программ.
***Ключевые слова*:** алгоритмизация перевода, перевод текста, активная грамматика, пассивная грамматика.

Глава 3. В.В. Миронов, А.К. Розанов
АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ КОНТРОЛЯ И ПОИСКА ДАННЫХ
Часть 1
Аннотация. Первая часть посвящена системе контроля знаний по морфологии Salvinia Examiner, являющейся вспомогательным средством для обучения морфологии русского языка. Этот комплекс обучающего и контролирующего программного обеспечения позволяет решать следующие задачи в обучении: формирование умения определять грамматическую информацию для написанных слов (распознавать число и падеж имён существительных, время глагола и т.д.); формирование навыка построения требуемых форм слов (например, умения просклонять некое существительное).
***Ключевые слова*:** контроль знаний по морфологии.

Часть 2

Аннотация. Во второй части разработана информационная система, реализующая русско-английский пословный перевод математических терминов. В основе системы лежит оригинальный русско-английский словарь объемом 40 000 слов и словосочетаний с широким охватом специальных математических терминов. Представлены алгоритмы, позволяющие преобразовать форматированный текст документа (или фрагмента) Microsoft Word в формат HTML. Предложенная информационная система может использоваться как для реализации электронных словарей, так и для более широкого круга задач, требующих преобразования форматированного текста в Word, в HTML или иное подобное HTML представление. Гибкость системы позволяет трансформировать словарь в математическую энциклопедию.

Ключевые слова: русско-английский словарь, информационно-поисковая система, перевод текста.

Глава 4. С.А. Нелюхин при участии В.В. Миронова
ОПИСАНИЕ СИСТЕМ КОМПЬЮТЕРНОЙ АЛГЕБРЫ *MAPLE* И *WXMAXIMA*

Аннотация. Системы компьютерной алгебры (СКА, англ. Computer Algebra System, CAS) являются одними из необходимых компонентов информационных технологий, используемых в образовании. СКА могут применяться для решения широкого круга задач – от нахождения решения простейшего линейного уравнения в символьном виде до исследования серьезной научной работы, в которой результат желательно получить в аналитическом виде. Поэтому основной задачей любой СКА является представление результата в символьном (аналитическом) виде.

Ключевые слова: системы компьютерной алгебры, результаты в символьном виде.

Глава 5. С.А. Нелюхин при участии В.В. Миронова
ПРИМЕНЕНИЕ СИСТЕМ КОМПЬЮТЕРНОЙ АЛГЕБРЫ *MAPLE* И *WXMAXIMA* ДЛЯ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ

Аннотация. В процессе изучения дисциплины естественно-математического цикла одной из форм обучения является самостоятельное освоение студентами новых информационных технологий, в частности современных систем компьютерной математики (СКМ). Совместно с традиционными формами и методами обучения (лекции, упражнения, лабораторные работы) и в дополнение к основным современным языкам программирования (Pascal, C# и т.д.) на основе данного пособия студенты могут самостоятельно проводить анализ поставленной задачи в определенной СКМ и решать задачу.

Ключевые слова: система компьютерной математики.

Глава 1

ИНФОРМАТИЗАЦИЯ ОБРАЗОВАНИЯ: ДОСТИЖЕНИЯ И ПРОБЛЕМЫ

Введение

Данная глава написана в развитие работы Valentin V. Mironov, Specificity of communication teacher-student in informational and technological era // SHS Web of Conferences, **9**, 02028 (2016) DOI: 10.1051 /shsconf/ 2016 EEIA, 2016 2 2902028/

Глава существенно опирается также на материал и развивает идеи, высказанные в работе [1].

В данной главе проведен анализ проблемы информатизации образования и угроз, которые информатизация образования способна нести в себе. Рассмотрены принципы и границы информатизации и тех основ образования и воспитания, что стоят за всем процессом информатизации.

1. Общие положения

Напомним, что информатизация (Федеральный закон "Об информации, информационных технологиях и о защите информации" от 27.07.2006 N 149-ФЗ принят Государственной Думой 8 июля 2006 года, одобрен Советом Федерации 14 июля 2006 года) – это "организационный социально-экономический и научно-технический процесс создания оптимальных условий для удовлетворения информационных потребностей и реализации прав граждан, органов государственной власти, органов местного самоуправления, организаций, общественных объединений на основе формирования и использования информационных ресурсов".

Применительно к образованию информатизация образования – это целенаправленный, созидательный, творческий, создающий новое, креативный процесс обеспечения сферы образования методологией, технологией и практикой разработки оптимального использования средств информационной и компьютерной техники, применяемых в современных, щадящих здоровье условиях. Это процесс, ориентированный на достижение психолого-педагогических целей обучения, развития человека, включающий в себя в качестве подсистем как непосредственно обучение, так и воспитание.

Суть процесса информатизации изложена в Федеральном законе «Об образовании в Российской Федерации» от 29.12.2012 N 273-ФЗ. Приведем отрывок полностью.

Статья 16. Реализация образовательных программ с применением электронного обучения и дистанционных образовательных технологий

1. Под электронным обучением понимается организация образовательной деятельности с применением содержащейся в базах данных и используемой при реализации образовательных программ информации и обеспечивающих ее обработку информационных технологий, технических средств, а также информационно-телекоммуникационных сетей,

обеспечивающих передачу по линиям связи указанной информации, взаимодействие обучающихся и педагогических работников. Под дистанционными образовательными технологиями понимаются образовательные технологии, реализуемые в основном с применением информационно-телекоммуникационных сетей при опосредованном (на расстоянии) взаимодействии обучающихся и педагогических работников.

2. Организации, осуществляющие образовательную деятельность, вправе применять электронное обучение, дистанционные образовательные технологии при реализации образовательных программ в порядке, установленном федеральным органом исполнительной власти, осуществляющим функции по выработке государственной политики и нормативно-правовому регулированию в сфере образования.

3. При реализации образовательных программ с применением исключительно электронного обучения, дистанционных образовательных технологий в организации, осуществляющей образовательную деятельность, должны быть созданы условия для функционирования электронной информационно-образовательной среды, включающей в себя электронные информационные ресурсы, электронные образовательные ресурсы, совокупность информационных технологий, телекоммуникационных технологий, соответствующих технологических средств и обеспечивающей освоение обучающимися образовательных программ в полном объеме независимо от места нахождения обучающихся. Перечень профессий, специальностей и направлений подготовки, реализация образовательных программ по которым не допускается с применением исключительно электронного обучения, дистанционных образовательных технологий, утверждается федеральным органом исполнительной власти, осуществляющим функции по выработке государственной политики и нормативно-правовому регулированию в сфере образования.

4. При реализации образовательных программ с применением электронного обучения, дистанционных образовательных технологий местом осуществления образовательной деятельности является место нахождения организации, осуществляющей образовательную деятельность, или ее филиала независимо от места нахождения обучающихся.

5. При реализации образовательных программ с применением электронного обучения, дистанционных образовательных технологий организация, осуществляющая образовательную деятельность, обеспечивает защиту сведений, составляющих государственную или иную охраняемую законом тайну.

Вообще термин «информатизация образования» имеет широкое распространение применительно и к различным вариантам, технологиям использования компьютеров, компьютерных сетей и соответствующих информационных технологий в управлении образовательными учреждениями и системами таких учреждений.

Процесс информатизации инициирует:

1. Обновление методов и механизмов административного управления всей системой образования на основе достаточно полных банков данных научной, медицинской и педагогической информации, сопровождающих информационных и методических материалов, аккумулирующих прежний опыт работы, с использованием современных коммуникативных сетей.

2. Обновление собственно методологии как системы методов, общей стратегии процесса отбора внутреннего содержания, внешних методов и методов организации разнообразных форм обучения и воспитания, которое соответствует задачам развития личности обучаемого и задачам развития общества, в которое погружена личность, и все это в быстроменяющихся условиях информатизации.

3. Ввиду лавинообразного потока общей информации чрезвычайно важно создать такую систем обучения, которая развивает природный потенциал ученика, заставляя его добровольно, самостоятельно получать, систематизировать, апробировать знания, осуществлять информационную, учебную, экспериментальную, исследовательскую работу, направленную на реальное производство и науку.

4. Создание обучающих, тестирующих, диагностирующих методик и систем контроля с обратной связью и максимально объективной оценки уровня и объема знаний учащихся.

Сам процесс информатизации и его последствия (впрочем, как и во всякой большой системе) глубоко противоречивы. Как следствие, его протекание нуждается в перманентном осмыслении, оценке его параметров, экстраполяции и аппроксимации его последствий в научном, техническом, социальном и философском аспекте.

Говоря сжато, если угодно, в узком смысле, информатизация образования – это внедрение в систему образования микропроцессорной техники, а также всей сопутствующей ей информационной продукции и технологий. Цель информатизации образования (как и всего процесса информатизации жизни) проста – создать новую, информационно-коммуникационную, если хотите, виртуальную среду, в которой человек, или группа людей, или общество в целом могли развиваться максимально быстро с наивысшими результатами. И здесь (см. выше TOP 10 ведущих компаний мира по проблеме искусственного интеллекта, в числе которых нет, увы, российских компаний и университетов), образно говоря, кто не успел, тот опоздал.

Анализ показывает, что для достижения целей информатизации необходимо:

1. Развивать информационную инфраструктуру, которая связала бы в одну сеть образовательные учреждения отдельных регионов или страны в целом, причем на основе уже привычных для пользователей информационных и телекоммуникационных технологий.

2. Создать систему постоянного обновления содержания информационной среды, обновлять образовательные ресурсы с учетом их

дальнейшего применения в учебном процессе и нового обновления, создавая тем самым перманентный процесс обновления.

3. Создать целенаправленную систему подготовки и переподготовки кадров, преподавателей, методистов, руководителей учебных заведений, направленную на эффективное овладение ими информационно-коммуникационных технологий и образовательных ресурсов.

2. Историческое осмысление этапов и проблем информатизации

Мы не первые, кто задумался над проблемой эффективности информатизации образования - явления для педагогической практики достаточно нового, как и сам компьютер.

Практика осмысления роли техники в жизни человека и общества начинается с работ Н.Л. Бердяева, К. Ясперса, Дж. Тойнби, Н. Винера, Ж. Эллюля, Л. Мэмфорда, Н.М. Чуринова, И.А. Негодаева. И вот уже период мощного научно-технического прогресса называют постиндустриальной эпохой (Д. Белл), информационным обществом (Масуда), третьей волной индустриализации (А. Тоффлер) и даже новой цивилизацией (Р.Ф. Абдеев). Становится ясно, что традиционное общество меняется коренным образом.

Противоречивость отношений людей в техническом обществе изучали российские ученые А.Н. Леонтьев, Б.Ф. Ломов, В.Л. Николаев, В.Ф. Рубахин, изучали и обозначили социальные и психологические особенности развития человека в системе «человек - машина».

Гносеологическая проблема определения границ и меры информатизации образования решалась в рамках так называемого Шведского проекта ОТТ – «Образование-Труд-Техника», действовавшего с 70-х годов прошлого века до начала века XXI. Проект был направлен на изучение проблем и противоречий информатизации сферы социального страхования. Основной вывод состоял в том, что в обыденном сознании гипотеза об оптимальности новых технологий в любых сферах быстро завоевывала позиции и становилась для людей неоспоримой аксиомой. Проблемы появлялись гораздо позже и носили глубинный характер, не заметный на первых этапах.

Теоретическими предпосылками для описания границ и меры информатизации образования стали работы К. Шеннона, А.С. Нариньяни, Дж. Экклза, Н. Вайсштейна, А.Л. Самсонова, И.М. Амосова, Б.Ф. Ломова и др. Главное – польза информатизации несомненна, но отчетливо просматривается тенденция, согласно которой и технический базис информатизации далеко не совершенен, и последствия его не изучены, возможно, полны драматизма.

Существуют серьезные исследования в области математической логики, которые доказывают теоретическую невозможность создания искусственного интеллекта (ИИ), обладающего сознанием в человеческом его понимании и наполнении, и, как следствие, отсутствие катастрофических угроз для человека со стороны ИИ (группа «оптимистов» – Дж. Лукас, Е.Ефремов, Р. Пенроуз, В.А. Глазунов, П. Беиакерраф и др.). Одновременно функционирует

целая группа представителей интенциональной теории познания (группа «пессимистов» – Д. Даннет, Дж. Серль и др.), которая доказывает обратное и стремится создать всеобъемлющий ИИ. А значит, перед человечеством встанет нешуточная угроза со стороны ИИ, который в какой-то момент может спросить: «А зачем здесь человек, если он более чем наполовину и не человек уже, а Кибернетический организм?». Парадокс в том, что остановить эти исследования на законодательном уровне невозможно, как не удалось остановить в свое время атомный и другие смертельно опасные проекты. Вторая сверхугроза состоит в том, что «расстояние» от разработок до применения в новых технологиях может быть очень малым. Поневоле и верующий, и атеист воскликнут: «Храни нас Бог!».

Осмыслению свойств человека, как носителя знаний посвящены работы В.В. Свидерской, в которых описаны подуровни деятельности и существования человека (биологический, психологический, социальный, нравственный, этический). Работы М.А. Петрова посвящены осмыслению взаимоотношения понятий «информация» и «знание».

Вопросам описания гносеологической границы и меры информатизации образования посвящены исследования форм познания (Ю.М. Сердюков, В.Н. Шестаков). Изучалась также проблема неявных и скрытых знаний, составляющих значительный человеческий опыт (М. Полани, Р.Дж. Стерпберг, Бу Геранзон, Н.Т. Абрамова, Н. Хомский).

Описанию социальной границы и меры информатизации посвящены исследования В.Н. Шестакова. В этом контексте Г. Бейтсон предложил логическую иерархию категорий обучения, позволившую, в частности, определить предельный уровень информационных систем. Работы по исследованию коммуникативности индивидов позволили выделить ее цифровой и аналоговый тип (Б. Рассел, П. Вацлавик и др.). Изучались также методы и способы моделирования стратегий поведения и мышления человека и групп людей (Р. Дилтс).

В целом можно сказать, что библиография по вопросам информатизации жизни и, в частности, образования весьма обширна, но устойчивой, общепринятой ее классификации на сегодняшний день нет.

3. Кризисные явления

Информатизация общества является тем контекстом, на фоне которого функционирует и развивается информатизация образования. Последнее, как и все общество, регулярно переживает системные кризисы как внешнего, так и внутреннего свойства. Так, для современного, технологичного и технократичного образования вновь актуальны вопросы: «зачем учить?», «кого учить?», «в какой мере учить?», «как учить?», «за чей счет учить?» и другие подобные.

Надо отметить, что в эпоху кризисных явлений, остро переживаемых нашей страной, в поисках выхода из этих явлений жгучие вопросы образования не просто обсуждаются, а сопровождаются реальным

реформированием образования, которое, зачастую, как видно из итогов реформ, не отвечает дальновидности.

Нынешние тенденции реформирования образования приводят к смене самой образовательной парадигмы, основ функционирования. Новой парадигмой образования становится коммерциализация образования, которая с неизбежностью вытесняет воспитание из учебного процесса. Процесс образования становится «образовательной услугой» с преобладанием компьютеризированных средств обучения, совершенно или частично заменяющих учителя, преподавателя и его тонкий общественно-полезный и личный труд и заменяющих его на услуги машины (в дальнейшем искусственного интеллекта) (см. Федеральный закон «Об образовании в Российской Федерации» от 29.12.2012 N 273-ФЗ.).

Законный вопрос о границах информатизации в образовании возникает в связи с масштабностью внедрения информатизации во всех сферах и областях деятельности человека и ученика, в частности.

С другой стороны, очевидно, что опасно осуществлять серьезные изменения, не оценив еще до эксперимента со всех сторон последствий нововведений и не спрогнозировав «лекарства от болезней». Вот почему эта повсеместная компьютеризация и информатизация учебных заведений, это быстрое развитие систем дистанционных технологий обучения должны быть подвергнуты критическому анализу, тем более что традиционная педагогическая практика входит в противоречие с новыми методами.

При выявлении меры информатизации в образовании прежде всего необходимо выделить три философских аспекта проблемы: онтологический, гносеологический, социальный. Затем можно перейти ко второй очереди актуальных вопросов – к финансовым и техническим ограничениям [2].

Остановимся коротко на постановке этих аспектов, наиболее важных с общефилософских и прагматических позиций.

При определении онтологической границы информатизации в образовании необходимо ответить на вопрос, с какими сущностными проблемами на пути решения образовательных задач (обучения и воспитания) средствами информационных технологий сталкиваются ныне учащиеся и специалисты или столкнутся в ближайшем будущем.

Изучение гносеологической границы информатизации образования продиктовано современными научно-техническими достижениями в исследовании различных, в частности нерефлексивных, форм познания. В этом случае встает вопрос построения эффективных (по различным критериям) моделей представления знаний, ответ на который просто необходим для разработки информационных образовательных систем.

Изучение социальной границы информатизации образования неразрывно связано с уникальной спецификой человеческой природы.

Финансовые вопросы, точнее отсутствие финансирования разработки, производства и внедрения информационных проектов, способны перечеркнуть все благие начинания и разработки в области информатизации

образования, с чем наша школа постоянно сталкивается. Вопрос этот ввиду перманентного кризиса сложный и не всегда разрешимый.

В этом же ряду стоит и проблема технических ограничений. Все технические задачи решаются двумя способами:

1) Есть задача, под нее выделяются деньги, материалы и люди;

2) Есть такая-то сумма денег, за эти деньги нужно построить систему.

К сожалению, информатизация образования в нашей стране идет по второму пути и конца этому не видно…

Таким образом, обоснована задача исследования границ информатизации образования, ибо в развивающихся процессах информатизации образования, объективно присутствуют противоположности, противоречия, а за ними – возможные негативные последствия. И опыт преподавания, опыт внедрения технологий не может быть здесь переоценен.

4. Практический опыт

Анализ опыта внедрения и эксплуатации информационных систем, применение современных средств информационно-компьютерных технологий (ИКТ) в процессе обучения приводят к выявлению целого ряда объективно присущих процессу негативных последствий.

Во-первых, чаще всего одним из преимуществ обучения с использованием ИКТ называют индивидуализацию обучения. Однако в этом существуют крупные недостатки, которые связаны с тотальной индивидуализацией. Индивидуализация сокращает и так ограниченное в учебном процессе живое общение учителей и учеников, предлагая последним общение в виде "диалога с компьютером". Это приводит к тому, что орган объективизации мышления человека - речь выключается из процесса обучения и ученик не получает практики общения в виде диалога с учителем, что ведет к атрофированию умения грамотного формулирования мысли на профессиональном языке. Попутно исключаются два (из трех) вида памяти – слуховая память и моторная память, когда ученик произносит или записывает то, что нужно запомнить. Зрительная же память чрезвычайно перегружается.

Во-вторых, недостатком использования средств ИКТ в образовании является переход от информации, распространяющейся в системе обучения, к самостоятельным профессиональным действиям, т.е. от знаний на страницах учебника к системе практических действий, имеющих кардинально новую логику. Это классическая проблема применения знаний на практике, формальных знаний, а на психологическом языке - проблема перехода от мысли к действию.

В-третьих, проблемы возникают в результате применения современных средств ИКТ, предоставляющих свободу в поиске и использовании информации. Часто обучаемые, психика и внимание которых неустойчивы, отвлекаются от поиска и изучения материалов из-за запутанных и сложных методов представления информации и ее различных несоответствий.

Огромные объемы информации, предоставляемые некоторыми средствами информатизации, такими как электронные справочники, энциклопедии, Интернет-порталы, также могут отвлекать внимание в процессе обучения.

Кроме того, использование информационных ресурсов, опубликованных в сети Интернет, часто приводит к отрицательным последствиям. Чаще всего при использовании средств ИКТ срабатывает свойственный всему живому принцип экономии сил: заимствованные из сети Интернет готовые проекты, рефераты, доклады и решения задач из школьных учебников стали сегодня в школе уже привычным фактом, не способствующим повышению эффективности обучения и воспитания школьников. Работа по образцу стала главным методом обучения в школе или вузе, решение творческих задач вызывает у учащихся непреодолимые трудности.

К примеру, статистика входного контроля на всем первом курсе в Рязанском государственном радиотехническом университете показала, что график функции $y = 2\sin 2x$ строят 9 учеников из 10, а график функции $2x = \sin 2y$ строит 1 ученик из 10. При этом никто даже и не спрашивает, что здесь функция, а что аргумент функции, а без такого вопроса задача становится некорректной, а ее решение – недетская проблема!

В некоторых случаях использование средств информатизации образования лишает учеников возможности проведения реальных опытов своими руками, что отрицательно сказывается на результатах обучения.

И, наконец, нельзя не видеть или вовсе забывать о том, что чрезмерное и не оправданное использование большинства средств информатизации негативно отражается на здоровье всех участников образовательного процесса, как учеников, так и учителей.

Анализ показывает, что основные трудности учителя по применению ИКТ в преподавании предметов состоят в следующем:

1. Проблемы с поиском информации. С одной стороны, информации в сети Интернет предостаточно, т.е. доступ учеников к материалам для подготовки к уроку (найти портреты писателей, иллюстрации к их произведениям и т.д.) обеспечен всем. Однако в Интернете встречаются несоответствия и даже ошибки в материалах. Поэтому важно точно знать, с каких сайтов брать информацию.

2. Для подготовки к экзаменам по предметам, например по русскому языку, легко найти тестовые задания по определенной теме. Но к этим тестам легко найти ответы, что заставляет учителя лишний раз разговаривать с учениками на тему самодисциплины и честного определения уровня знаний.

3. Важно отметить, что учителей обязали вести электронный журнал (быстрый, удобный доступ к оценкам и домашнему заданию), который значительно усложнил их работу (при неизменной заработной плате), т.к. заполнение электронного журнала намного дольше по времени, чем бумажного, без учета проблем с техникой.

4. Вообще на федеральном уровне отсутствует продуманная и обоснованная психоэмоциональными закономерностями учеников и преподавателей, поэтапность внедрения ИКТ на всех уровнях.

5. Границы информатизации или вне процесса

В последние годы в связи с повсеместным развитием и внедрением информационных технологий в экономические, социальные, политические, научные, образовательные сферы деятельности человека проводятся настойчивые попытки внедрить эти *технологии* и в такую тонкую «материю», как процесс общения учителя и его ученика. Проблема, на наш взгляд, состоит в том, что этими новыми компьютерными технологиями хотят ограничить или сопроводить *весь процесс обучения*, который, вне всякого сомнения, неотделим от *педагогики*!

Рассмотрим проблему подробнее.

Действительно, педагогика [греч. *paidagogike*] – это наука о воспитании и обучении человека; технология [греч. *techne* – искусство, мастерство; и *logos* – слово] – это 1) совокупность методов обработки, изготовления, изменения состояния, свойств, формы сырья, материалов в процессе производства; 2) наука о способах воздействия на сырье, материалы, полуфабрикаты соответствующими орудиями производства; инновация [англ. *innovation*] – нововведение, новое явление [3].

Рассматривая ученика высшей школы как единство будущего профессионала и личности, тут же впадаем в противоречие в вопросах применимости педагогических и информационных технологий, в частности информатизации. В самом деле рассмотрим человека, как профессионала в процессе обучения. На этой стадии можно говорить о технологии обучения как системе методов обучения (конкретная, профессиональная методика преподавания) или как о науке об общих закономерностях обучения (психолого-философские методы в обучении).

В Рязанском государственном радиотехническом университете (который представляют авторы) – это система стандартных воздействий на студента: лекции и семинары. Роль индивидуальности профессора здесь не заменить никакими технологиями, достаточно задаться вопросом о скрытом знании или о невербальных формах передачи знаний. В этом контексте мы вспоминаем выдающихся профессионалов, работавших на математическом факультете МПГУ, – проф. Лемлейна В.Г., проф. Аронова Р.А., академика РАН, академика РАО, проф. Матросова В.Л., проф. Черкасова Р.С. и многих других учителей. С другой стороны, в РГРТУ создана целая система с обширной базой данных, включающая в себя типовые задачи и расчеты, лабораторные и контрольные работы, консультации, в том числе компьютерные и дистанционные (вот где простор для технологий!).

Как всякая технология, эта конкретная технология обучения предмету имеет нижнюю и верхнюю границы применимости и трудности реализации.

Анализ проведенных массовых обследований и тестирований в рамках рязанских школ, университетов, а также опросов учителей – слушателей

РИРО (Рязанский институт развития образования) и преподавателей рязанских вузов показывает, что справедливы следующие тезисы (если хотите, аксиомы информатизации).

ТЕЗИС 1. Любая технология обучения в системе «учитель-ученик» в принципе ориентирована на усреднение, массовое тиражирование.

ТЕЗИС 2. Не существует технологий «производства» выдающихся специалистов ни в одном из видов знаний.

Ориентация на «середняка», производство специалиста среднего уровня знаний – вот реальный выход (а может, и смысл?) всех технологий высшей школы последнего времени, в том числе информатизации образования. Частично разрушить верхнюю грань (или сдвинуть ее «вправо») применимости технологии обучения может разве что одна импровизация, конкретный подход к конкретному студенту (но кто же даст столько часов в высшей школе?). Таким образом, уже на первой стадии общения пары «учитель-ученик» имеет место следующий тезис.

ТЕЗИС 3. Всякая технология обучения, включая информационные технологии, в системе «учитель-ученик» внутренне (неустранимо) противоречива.

Совершенно неприемлемым представляется технологический подход ко второй диалектической грани системы «учитель-ученик» – это воспитание личности, порядочного человека, патриота Отечества. Последнее, кстати, неотъемлемо от категории «взаимности», о чем наглядно свидетельствует массовый выезд российских специалистов высшего класса за пределы Отечества (при полном спокойствии ответственных чиновников, уверенных, по-видимому, что бесконечно «может собственных Платонов и быстрых разумов Невтонов российская земля рождать»).

Обратимся к авторитетным мнениям в этих противоречивых вопросах – Ж. - Ж. Руссо, Ф.М. Вольтеру, Л.Н. Толстому, В.А. Сухомлинскому. Центральное звено их педагогических систем – уникальность, неповторимость личности и, как следствие, неповторимость педагогического воздействия на нее и, значит, неприемлемость технологий воспитания вообще (с другой стороны, император Нерон, или Чингисхан, или И.В. Сталин могли бы здесь предметно поспорить).

Одновременно существует идея унифицированного, технологического воздействия на личность, восходящая к взглядам и учениям Дж. Локка (теория «ученика, как чистого листа»). В советское время эта идея получила развитие в работах педагога и писателя А.С. Макаренко и была востребована временем. Работы нашего земляка академика И.П. Павлова по анализу высшей нервной деятельности и системе обучения животных стали для этих практик «научной» основой.

Этот технологический подход к личности явился одной из причин провала в решении фундаментальной задачи воспитания «нового человека», провозглашенной в Программе КПСС 1961 г. и в ее Модернизации в 1986 г.

ТЕЗИС 4. Не существует эффективных педагогических технологий по воспитанию позитивной человеческой личности.

Дополнительно обосновывая этот тезис, укажем, в частности, на тот неоспоримый факт, что одним из элементов воспитания в системе «учитель-ученик» является доверительность. Но технология, разговаривающая сразу со всеми, по самой природе своей есть антидоверительность.

Таким образом, не существует эффективных педагогических технологий по воспитанию уже потому, что справедлив следующий тезис.

ТЕЗИС 5. Нельзя доверительно разговаривать с массами.

Еще более спорной представляется идея инновационных (слово, столь любимое некоторыми чиновниками, что его следовало бы выбить золотыми буквами на фронтоне их здания – «Нынешнее поколение российских людей будет жить при инновационном капитализме!») педагогических технологий. В самом деле, такие технологии могут возникнуть там и тогда, где и когда появится новая система человеческих ценностей (как это было в 1917 г.). Однако мировой и отечественный опыт показывает, что такой жизнеспособной системы на сегодняшний день нет.

В такой ситуации по продвижению информационных технологий в систему «учитель-ученик» неизбежно возникают извечные русские «проклятые» вопросы: Кто виноват? Что делать? В попытке ответить на них отчетливо просматриваются конъюнктурные, политические мотивы, желание заменить сложную, финансово емкую работу громкой «научной» фразеологией.

Одновременно, говоря о специфических аспектах применимости на разных стадиях обучения технологий в общении учителя и ученика, нельзя «вместе с водой выплеснуть и ребенка»! Другими словами, каковы общие (присущие процессу обучения в массе своей) проблемы общения учителя и ученика, каковы конструктивные особенности технологий ранних стадий обучения? Рассмотрим кратко и эти, отнюдь не риторические, вопросы.

Что понимать под педагогической технологией в системе «учитель-ученик»? Педагогическая технология – это достаточно общая система компонентов педагогической и образовательной деятельности в их определенной временной последовательности, направленная на достижение целей обучения, воспитания и общего развития учащегося.

6. Снова опыт

Какие же специфические черты технологий возможно и необходимо, на наш взгляд, внедрять в систему «учитель-ученик» в современную эпоху формирующегося единого образовательного пространства?

Это прежде всего:

- унификация образовательных стандартов при определенной их внутренней вариативности и альтернативности,
- целостный подход, системность и диалектическая взаимосвязь отдельных компонент системы,
- опора на светскость и демократизацию, на гуманизацию образования,
- ориентация на отдельную личность, а не на массовый «продукт»,

- современная информационная и технологическая подготовка воспитательных и педагогических кадров,

Выстраивая самые общие ориентиры педагогических «технологий» в системе «учитель-ученик», необходимо предусмотреть в них различные уровни:

- концептуальный,
- диагностический,
- целевой,
- информационно-содержательный,
- методический,
- аналитический,
- контрольный,
- корректирующий.

Остановимся на этой специфике и их уровнях подробнее.

О каких специфических чертах технологий в образовании в широком концептуальном смысле можно говорить, на наш взгляд?

Унификация образовательных стандартов при определенной их внутренней вариативности и альтернативности в нынешнем их состоянии не выдерживает критики хотя бы уже потому, что изменение образовательных стандартов происходит чуть ли не ежегодно. При этом следующий стандарт зачастую опровергает предыдущий либо наполняется его неоправданными и объемными формальными требованиями к сопровождению стандартов.

Целостный подход, системность и диалектическая взаимосвязь отдельных компонент системы. Этот тезис включает в себя понимание системности как направление методологии научного познания, в основе которого лежит рассмотрение объекта как системы: целостного комплекса взаимосвязанных в их естественности элементов.

Целостность, позволяющая рассматривать одновременно систему как единое целое и в то же время как подсистему для вышестоящих уровней.

Иерархичность, то есть наличие множества (по крайней мере, двух) элементов, расположенных на основе подчинения элементов низшего уровня элементам высшего уровня. Реализация этого принципа хорошо видна на примере любой конкретной организации высшей школы, к примеру. Как известно, любая такая организация представляет собой взаимодействие двух подсистем: управляющей и управляемой. Одна подчиняется другой.

Структуризация, позволяющая анализировать элементы системы и их взаимосвязи в рамках конкретной организационной структуры. Как правило, процесс функционирования системы обусловлен не столько свойствами её отдельных элементов, сколько свойствами самой структуры.

Множественность, позволяющая использовать множество кибернетических, экономических и математических моделей для описания отдельных элементов и системы в целом.

Опора на светскость и демократизацию, на гуманизацию образования. Особо отметим то положение, согласно которому правовое регулирование процессов в сфере образования основывается на принципе светского характера образования во всех государственных организациях, занимающихся образовательной деятельностью (п. 6 ч. 1 ст. 3 Закона «Об образовании в Российской Федерации»). По сравнению с ранее действовавшим Законом «Об образовании в Российской Федерации» от 1992 г. список государственных организаций, в которых действует этот принцип светскости, в новом законе существенно расширен. Если ранее светский принцип устанавливался в отношении государственных и муниципальных образовательных учреждений, то теперь он распространяется на «государственные, муниципальные организации, осуществляющие образовательную деятельность».

Во исполнение поручения Президента Российской Федерации от 2 августа 2009 года № Пр-2009 в 2009–2011 годах в школах ряда субъектов Российской Федерации был введен и апробирован учебный курс «Основы религиозных культур и светской этики» (ОРКСЭ), включающий основы и догматы православной культуры, исламской культуры, буддийской культуры, иудейской культуры, других мировых религиозных культур и, что особенно важно, основы светской этики. С сентября 2012 года этот курс преподается в четвертом и пятом классах во всех государственных и муниципальных общеобразовательных учреждениях России.

Ныне практика преподавания ОРКСЭ закреплена на уровне федерального закона. (Ст. 87 Закона «Об образовании в Российской Федерации», который содержит нормы, касающиеся преподавания учебных курсов, направленных на получение знаний о нравственных принципах и традициях православия, а также иных мировых религий).

Практика показывает, что иногда служители церкви стремятся к более полному и более широкому их присутствию в системе образования, нежели это предусмотрено законом. С другой стороны, и школа не должна ограничивать тягу учащихся к церкви и, более того, должна приветствовать обращение учеников к истории и традициям церкви и поощрять (а не запрещать) посещение ими храмов. В этой работе чрезвычайно важны мера, постепенность и такт. Вот только где найти умных духовных наставников?

Ориентация на отдельную личность, а не на массовый «продукт». Это то, отсутствием чего так больна нынешняя русская школа: она перестала основное внимание уделять талантливым ученикам, если хотите, гениям, на них не хватает ни времени, ни сил, а их подготовка стала уделом отдельных педагогических «фанатиков», готовых работать даром, «за идею». Это положение нужно в корне изменить. Как? Ясно, как: выделять под них дополнительные часы и платить за дополнительное обучение. Контроль возложить на администрацию школы или вуза, а критерий качества работы один – результаты образовательной и научной деятельности ученика. При этом надо понимать, что это - «венчурные» проекты: вложения не всегда оправдываются, но там, где оправдываются, отдача весьма велика! Конечно,

вопрос этот весьма сложный, может, поэтому и открыт, и долго будет открыт…

Современная информационная и технологическая подготовка воспитательных и педагогических кадров. Это большая проблема нашей школы, вызванная ее недофинансированием. Договорились уже до того, что «инженеров будем отправлять в школы на преподавательскую работу». Это нонсенс! Воспитанием и обучением, если не хотим отстать от мирового уровня в образовании навсегда, должны заниматься профессионалы.

Раскроем подробнее и уровни образовательной технологии.

Концептуальный уровень состоит в определении главных приоритетов, главных ориентиров технологий, признанных философских обоснований, научно-теоретических идей, системности основополагающих принципов, осмыслении и внедрении передового, но уже оправдавшего себя опыта, и, наконец, определении условий, в том числе материальных и финансовых!

Диагностический уровень предусматривает методы и способы постоянного мониторинга технологического процесса. Этот уровень подразумевает и соответствующий инструментарий, и квалифицированные кадры (психологи, методисты и т.д.).

Целевой уровень предполагает выявление и обоснование системного блока или множества целей и общих и конкретных задач образования и обучения, а также целей и задач воспитания (хотя последнее ныне все более и более изымается из практики высшей школы). Этот уровень предусматривает также креативность образования, его последовательность в достижении целей, а также методы воспитания духа конкурентной борьбы на рынке труда.

Информационно-содержательный уровень - это, как видится, формирование или наполнение общего педагогического образования конкретным (научным) знанием, включающим в себя как информационные данные, так и работы с ними.

Структурно к этому примыкает вопрос о конкретном содержании данных и их соответствии общей концепции. Этот уровень предполагает критический отбор фундаментальных знаний, обоснованную их направленность, прикладное значение (в том числе и главным образом профессиональное), широкий гуманитарный и культурный контекст.

Методический уровень включает в себя набор критериев оптимальности технологий и средств их реализации, выбор на этой основе методов, средств, приемов, их вариативность, взаимосвязь и логическую последовательность и отделенность (дифференцированность) методов.

Аналитический уровень состоит из систематического анализа всей образовательной деятельности, выявления правил и закономерностей процесса, анализа целевых установок, их обоснованности применительно к данной проблеме.

Контрольный уровень ныне предусматривает переход от традиционных отметок к рейтинговой системе оценки знаний с последующей апробацией накопленных знаний на решении практических задач. Проблема здесь в том,

что в российских реалиях высокие рейтинги не гарантируют успешного продвижения по профессиональному и административному пути.

Корректирующий уровень, основываясь на диагностирующем уровне, подразумевает внедрение методов достижения поставленных целей при возможности их обоснованной корректировки, корректировки как целей, так и методов их достижения. При этом глобальные критерии эффективности должны оставаться в силе.

В рамках этих общих рекомендаций под каждого студента (или однородную группу студентов или, более обще, учащихся) педагогами будут конструироваться конкретные технологии обучения. Да и тогда, как показывает практика, результат не гарантирован.

7. Скрытые знания

Остановимся совсем коротко еще на одной проблеме обучения, выпадающей из нынешнего образования при его тотальной информатизации, – это скрытое знание, обучение скрытому знанию.

Вообще знание [4] — это форма существования и систематизации результатов познавательной деятельности человека. Выделяют различные виды знания: научное, обыденное (здравый смысл), интуитивное, религиозное и др. Обыденное знание служит основой ориентации человека в окружающем мире, основой его повседневного поведения и предвидения, но обычно содержит ошибки, противоречия. Научное знание включает в себя логическую обоснованность, строгую доказательность, повторяемость результатов в схожих условиях, проверку достоверности, отсутствие систематической погрешности и разрешимость преодоления противоречий за счет нового знания.

Скрытые знания – это интуиция, опыт, секреты мастерства, ассоциации, навыки, мудрость, наконец. Тезис «know how» («знаю, как») не предполагает тезиса «знаю, почему так». Как следствие, скрытые знания порой трудно формализовать. Опыт показывает, что такими знаниями может владеть и учитель.

Впервые опыт использования скрытых знаний в производственных процессах описан в работе [5]. Оригинальный способ передать часть скрытых знаний – использование метода «storytelling» («рассказ о том, как это было»). Его применяют при создании корпоративных легенд и формализации части скрытых знаний.

Пример из личного опыта будет здесь уместен. Один из авторов (В.В.Миронов), еще будучи студентом, многократно прочитав и будто бы поняв, в конце концов, знаменитый пример акад. А.Н. Колмогорова всюду расходящегося ряда Фурье (пример *интегрируемой функции, ряд Фурье которой расходится почти всюду*), никак не мог понять: откуда этот пример взялся, к тому же он все время «ускользал» от понимания? Этот пример был уникален, не ясно было, как можно было до него додуматься? Читаешь книгу – ясно, книгу закроешь – не ясно.

После этого примера А.Н. Колмогорова (1921 г) долго было неизвестно, существует ли просто *непрерывная* функция с тем же свойством. Лишь в 1966 г. шведский математик Л. Карлесон (Lennart Carleson) доказал, что для всякой функции с интегрируемым квадратом ряд Фурье сходится почти всюду (с точностью до множества меры ноль) к самой функции (гипотеза Н.Н. Лузина, 1915 г.) и на непрерывные функции результат не распространяется.

- Простите, пожалуйста, в чем здесь дело? – набравшись смелости, спросил я о примере Колмогорова у одного известного профессора математики.

- А Вы попробуйте представить этот ряд сидящим на стуле, в кино, в зрительном зале среди своих друзей-рядов, которых Вы хорошо знаете, очень хорошо знаете. Посмотрите на них, на всех сразу, одним взглядом, уясните место каждого, – при этом профессор, положив мне руку на плечо, посмотрел так ободряюще, что я помню его взгляд и поныне. Помучившись с неделю-другую, я понял-таки пример А.Н. Колмогорова.

Что, способен компьютер подсказать и посмотреть на Вас также ободрительно, с верой в Ваши силы?

Заключение

Таким образом, авторы, развивая исследования, начатые в [1, 6], приходят к следующим выводам:

1. На ранних стадиях обучения (адекватные) компьютерные или информационные технологии полезны и, более того, необходимы, они позволяют обратиться сразу ко всей аудитории и заложить в учеников некий обязательный минимум, который хорошо усваивается ими при надлежащем усердии; в применении средств информатизации образования необходим взвешенный и четко аргументированный подход; компьютер – всего лишь инструмент, облегчающий и ускоряющий то, что ранее делалось без компьютера.

2. На высших стадиях обучения, при подготовке профессионалов необходимо уйти от общих технологий и их стандартов и под каждого учащегося или студента разработать индивидуальную «технологию» его обучения.

Впрочем, другие взгляды на проблематику информатизации образования и сопутствующих ей процессов тоже имеют место [7–18], иногда и в противовес изложенному в данной главе.

Литература к главе 1

1. Миронов В.В. Педагогические технологии в высшей школе: не абсурд ли? // Докл. Вторые рязанские педаг. чтен. Рязань, РГПИ, 1995.
2. Шестаков В.Н. Информатизация образования, ее мера и границы (социально-философский аспект): Дис. канд. филос. наук: 09.00.11 Красноярск, 2006. 180 с. РГБ ОД, 61:06-9/199.
3. Современный словарь иностранных слов.М.:Русское слово, 1993.

4. Современный философский словарь. СПб.: Академический проект, 2004. 864 с. ISBN 5-8291-0364-8
5. Н. Икуджиро, Т. Хиротака. Компания - создатель знания. Зарождение и развитие инноваций в японских фирмах / [Пер. с англ. А. Трактинского]. М.: ЗАО "Олимп-Бизнес", 2011. - 384 с.: ил.
6. Valentin V. Mironov, Specificity of communication teacher-student in informational and technological era // SHS Web of Conferences, **9**, 02028 (2016) DOI: 10.1051 /shsconf/ 2016 EEIA, 2016 2 2902028/.
7. Блауберг И.В., Садовский В.Н., Юдин Э.Г. «Системный подход в современной науке в кн.: Проблемы методологии системных исследований. М.: Мысль, 1970.С. 7-48.
8. О`Коннор Дж., Макдермотт Ион. Искусство системного мышления: необходимые знания о системах и творческом подходе к решению проблем (The Art of Systems Thinking: Essential Skills for Creativity and Problem Solving) (.М. Альпина Паблишер, 2011. № 978-5-9614-1589-6).
9. Jarkko Hautamäki, Finnish Lessons What Can the World Learn from Educational Change in Finland? (TCP, New York, 2011).
10. Elmore R. I Used to Think… And Now I Think. (ed.) (Harvard Education Press, Cambridge, 2011).
11. Sabel C., Saxenian A., Miettinen R., Kristensen P. H., Hautamäki J., Individualized Service Provision in the New Welfare State: Lessons from Special Education in Finland (Sitra Studies 62, Helsinki, 2011).
12. Thuneberg H., Hautamäki J., Ahtiainen R. et al. Conceptual Change in Adopting the Nationwide Special Education Strategy in Finland , JEC., 15, 1 (2014).
13. Wustenberg S., Stadler M., Hautamäki J., Greiff S., The Role of Strategy Knowledge for the Application of Strategies in Complex Problem Solving Tasks, TKL, 19 (2014).
14. Sing Kai Lo, Effect of biofeedback cycling training on functional recovery and walking ability of lower extremity in patients with stroke, KJMSc, 30, 1 (2014).
15. Sing Kai Lo, Electrophysical therapy for managing diabetic foot ulcers: A systematic review, IWJ, 10, 2 (2013).
16. Li J., Confucianism. In D. Pong (Ed.), Encyclopedia of modern China (MI: Charles Scribner's Sons, Detroit, 2009).
17. Li, J., & Hayhoe, R., Confucianism and higher education. In J. A. Banks (Ed.), SAGE encyclopedia of diversity in education, 1 (CA: Sage, Thousand Oaks, 2011)
18. De Bary, Wm. T., Confucian tradition and global education (CU Press, New York, 2007).

Глава 2

ВОПРОСЫ ИНФОРМАТИЗАЦИИ ОБРАЗОВАНИЯ В ПРОЦЕССЕ ОБУЧЕНИЯ СТУДЕНТОВ РУССКО-АНГЛИЙСКОМУ И АНГЛО-РУССКОМУ ПЕРЕВОДУ С АВТОМАТИЗИРОВАННЫМ КОНТРОЛЕМ ЗНАНИЙ ЧЕРЕЗ ПРИМЕНЕНИЕ КОМПЬЮТЕРНЫХ ПРОГРАММ, РЕАЛИЗУЮЩИХ АЛГОРИТМЫ АКТИВНОЙ И ПАССИВНОЙ ГРАММАТИК

Введение

Данная глава написана в развитие работы авторов Valentin V. Mironov, Alexandr I. Zavolokin and Aleksey K. Rozanov. Preparing electronic handbook for using active grammar during process of translation of technical texts into English. SHS Web of Conferences **9**, 02029 (2016).

Глава состоит из пяти частей и посвящена актуальным вопросам происхождения естественного языка, выведению грамматических категорий и новым методам обучения заинтересованных лиц (студентов, аспирантов, научных работников, везде далее – студентов) русско-английскому и англо-русскому переводу с использованием математических принципов и методов алгоритмизации перевода. В работе дается теоретическое обоснование применению активной и пассивной грамматик, показаны возможные варианты их практического использования в учебном процессе, в том числе и в особенности, с использованием компьютерных программ.

В первой и во второй частях акцент сделан на рассмотрении актуальных вопросов возникновения естественного языка, а также обоснован некий процесс выведения «всеобщей формулы» познания и языка.

Во второй же части выводится оригинальная «грамматическая формула» языка.

В третьей части дается теоретическое обоснование активной и пассивной грамматик применительно к рассматриваемым вопросам.

В четвертой части приведены и обоснованы адаптированные к практике практические методы обучения студентов владению активной грамматикой в процессе перевода с русского языка на английский.

В пятой части предложены и реализованы практические методы по обучению студентов англо-русскому переводу с автоматизированным контролем знаний на базе оригинальной компьютерной программы, реализующей алгоритм использования пассивной грамматики.

1. Возникновение языка и выведение «всеобщей формулы» познания и языка (ФПЯ).

В этой части работы будут изложены и объяснены с точки зрения диалектико-философских принципов:

1. Процесс зарождения языка, его становление, содержание и сущность.
2. Процесс выведения «всеобщей формулы» познания и языка (ФПЯ).

3. Пример практического применения ФПЯ в обучении студентов переводу с одного языка на другой.

Интеллектуальное богатство любого общества выражается в языке. Язык – это, прежде всего, слова, то есть минимальные значимые части речи, полноценно реализуемые только через употребление их в предложении. Поэтому данное исследование начинается с краткого анализа *слова*, затем *предложения* и, наконец, *языка* как одного из главных инструментов *познания*.

По форме существования *слово* есть значимый отрезок звука, создаваемого движениями органов речи одного человека и воспринимаемого органами слуха другого человека. *Слово* этимологически связано со словами *слух, словити,* (*словити* - значит «*говорить*»), то есть с понятием «*то, что можно слышать*» [1, с. 384, 385]. Таким образом, *слово* определяется тем, кто его произносит, временем (долготой) произнесения, характером произнесения и тем, кто воспринимает его. *Слова*, необходимо и строго определенным образом объединяемые в *предложения*, удовлетворяющие какие-либо человеческие интеллектуальные потребности, сводимые в конечном итоге к удовлетворению их материальных и духовных потребностей, образуют то *движение* человека, которое в целом мы можем назвать *языком*. Поскольку *язык* неоспоримо есть *движение*, то он должен иметь *начало и конец*. Решим задачу: найти *начало языка*.

Поскольку применительно к языку затронут термин *движение*, то следует кратко на этом термине и остановиться. *Движение* есть способ существования материи, её всеобщий атрибут [2, с. 138]. То есть оно, движение, есть «изменение вообще» [3, с. 563] и «не только перемена места; в надмеханических областях оно является также и изменением качества» [3, с. 568]. Самое простое движение представляет собой удаление и приближение объектов друг к другу, а это подразумевает существование таких понятий, как *пространство* и *время*. Все это значит, что источник и начало языка надо искать в комбинации следующих понятий: *материя, движение, объект, пространство, время*.

Отметим особо, что авторы не затрагивают в этом исследовании дискуссионные, «тонкие» вопросы существования и движения Духа, взаимоотношения Духа и материи, а также (что одному из авторов ближе всего) рассмотрение Духа как необычной формы необычной, «сознательной» Материи, Материи с нулевыми массами и бесконечными скоростями в пространстве, где бесконечность тождественна точке, где энергия тождественна времени, и Ее определяющим влиянием на «грубую» материю по непонятным доселе законам.

Человеческое общество прочно закрепило в звуковом языке источник и начало этого движения. Но прежде обратимся к весьма любопытному исследованию приматологов из Университета Южной Калифорнии, которые составили словарь жестов обезьян [4]. Характер поведения обезьян, их жестикуляция и мимика при общении с сородичами обнаруживают у них отношение к другим обезьянам только как ко второму лицу – на «ты» и при

непосредственном визуальном контакте. У обезьян не обнаруживаются жесты, относящиеся к третьему лицу при общении на «ты» друг с другом, то есть обезьяны не способны передавать знания друг другу относительно чего - то третьего и формировать *сознание*.

Абсолютное большинство людей первым словом в своей жизни произносят *мама* [5]. Это же слово *мама* есть обращение ко второму лицу, то есть к тому, кто тебя родил. Иначе говоря, *мама* есть имя того, с кем ребенок непосредственно контактирует. Таким образом, можно судить о возникновении *имени существительного*. От *мама* произошло слово *мать*, и оно означает «*родительница*» [1, с. 227]. Из этого же слова *мама* как производное развилось слово *материя* «*строительный лес, ткань*» [6, с. 181]. Ребенок поначалу все предметы нередко называет *мамой*, впоследствии он научается различать предметы и понимать, что не все они *являются родительницей* по отношению к нему. У ребенка начинает проявляться понимание сущности окружающих его вещей как результат познания. Иначе говоря, у ребенка уже сформированы зачатки понятий (в развитие этих заключений смотри ниже рис. 6).

Как известно, *слова* получают конкретное значение, обретают *контекст*, если они употребляются в *предложениях*, ибо только в предложениях слова раскрывают то или иное свое значение. Само по себе *имя существительное* мало что несет в себе конкретного. Конкретное значение в нем выражается через его предикат. Ребенок пока еще произносит только одно слово – *мама* или, реже, иное слово (*папа, баба, дай* и др). Это слово у него еще не развивается в полностью произносимое *предложение*. Однако, произнося слово «*мама*», ребенок выражает целое суждение; суждение, которое может означать примерно следующее: «*мама есть, существует …*», «*мама, дай* (*например, кушать*)», «*мама, у меня болит …*» и т.п. В этом случае слово «*мама*» уже в самом себе содержат свое же *бытие*, через посредство которого ребенок научается отличать маму от остальных объектов. Впоследствии это связка начинает произноситься.

Понятие *являться* выражается также такими словами, как *существовать, есть, быть, называться* и т.п. Ключевым словом и понятием здесь является слово *быть*, что изначально значило *расти, произрастать*, затем *существовать, наличествовать* [1, с. 45].

Таким образом, найден источник движения – *материя* (а человек материален) и *начало* этого самого движения уже некоторое время спустя после рождения человека.

Рассмотрим далее вопрос познания и зарождения языка. Диалектика говорит нам о нескольких уровнях процесса познания через отражение объективной действительности материального мира [7, с. 49-70]. Основываясь на них, выделим 4 этапа вывода ФПЯ:

1. Отрицание нулевого порядка «застывшей» материей самое себя через свое (внутреннее и внешнее) движение.

2. Отражение первого порядка и возникновение идеального (образного) мира, отрицание материи.

3. Отражение второго порядка (или самоотражение уже отраженного) и возникновение образного мышления (разума), отрицание несущественных моментов познаваемых объектов.

4. Отражение третьего порядка, отрицание образов и возникновение на их месте условных звуковых знаков и формирование языка. Возникновение сознания. Появление разных языков.

Приведем (для удобства изложения) некоторые сокращения, знаки и условные обозначения, используемые ниже в главе:

материя – латинская буква **М** от *лат*. **materia** материя, вещество, (первичное) начало,

движение – русская буква **Д** от *лат*. **motus** движение (но для отличия от **М** - (материя) будет использоваться русская буква,

объект – латинская буква **О** от *лат*. **objectum** объект, в свою очередь от *лат*. **objectus** лежащий (находящийся) впереди, противолежащий (в нашем случае субъекту),

субъект – латинская буква **S** от *лат*. **subjectum -** субъект, подлежащее,

активный – латинская буква **a** от *лат*. **āctīvus** - действенный, практический; деятельный; действительный,

страдательный – латинская буква **p** от *лат*. **passio** – страдательность,

субъект активный – латинские буквы **Sa**,

субъект страдательный – латинские буквы **Sp,**

сказуемое, предикат – латинская буква **P** от *лат*. **praedicatum** – сказуемое,

глагол – латинская буква **V** от *лат*. **verbum -** глагол,

глагол активный – латинские буквы **Va**,

глагол страдательный – латинские буквы **Vp**,

стрелки ↔, ↕ и другие означают отношения и переходы,

обозначения «'», «"» и «'"» верхним индексом означают отражения первого, второго и третьего порядка соответственно.

Материальный мир. Поскольку *материя* есть источник и носитель всякого *движения*, то поместим *материю* в основании этого самого *движения*, то есть внизу. Получаем схематическое отношение, изображенное ниже в виде диаграммы:

Д

↕

М

Материя (М) сама по себе не дана нам в идеальном, чистом виде. *Движение* же (самое простое – это приближение и удаление) возможно только в случае, когда существуют минимум два *объекта,* которые меняют свое местоположение относительно друг друга или его удерживают. А это в свою очередь необходимо подразумевает *пространство* и *время*, в течение которого объект преодолевает или удерживает расстояние. Таким образом, *материя распадается* на *объекты, предметы*, что само по себе тоже есть *движение*.

Слово *объект* образовано от лат. objektum, в свою очередь образованного от глагола objicere - «бросать вперед, навстречу» [1, с. 269]. Синонимичное ему русское слово *предмет* означает «*метать перед собой*» [1, с. 326].

В результате приходим к некоей «формуле» материального мира в виде:

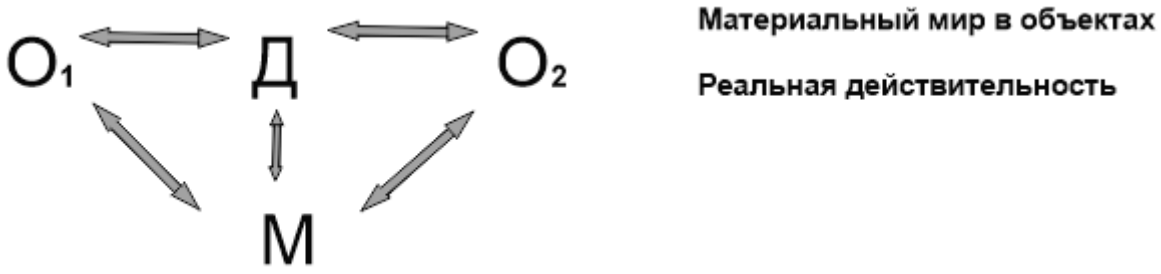

Рисунок 1. Материальный мир в объектах и реальная действительность

Объекты материального мира можно условно разделить на мертвые и живые. Из живых выделим те, что наделены физиологическим мозгом (без акцента на его интеллектуальные возможности). Ни одно животное без этого органа не существует. На этапах рассмотрения проблемы необходимы такие термины и понятия: *животное*, *человек*, *общество*, *отражение*, *отрицание*, через посредство которых будут выведены понятия *идея*, *знание*, *познание, язык* и *сознание*.

Отражение первого порядка. *Животные* (а человек как биологическая особь есть животное) в предлагаемой терминологии есть объекты. Они отличаются от неживого мира тем, что обладают способностью производить обменные процессы с другими объектами (это выражается, например, через дыхание или питание), посредством которых поддерживают в себе жизнь, и отражать в мозгу с помощью органов чувств окружающий мир. Главными органами чувств в процессе отражения в данном случае выступают зрение, слух, обоняние, осязание, вкус. А то, что зрение есть главный чувственный орган, посредством которого животное отражает в мозгу объективную действительность и получает знания о ней, говорит наличие в языках глагола *явить*(*ся*), *являть*(*ся*), то есть *делать*(*ся*) *видимым*, *явным* [1, с. 505-506].

Отражение объектов в мозгу животного имеет форму *идей* или *образов* (рис. 1), которые и образуют *идеальный мир*. Этимология слова *идея* восходит к слову *видеть* и означает *образ* (см. *идея* [6, с. 102] и *образ* [1, с. 267]).

Таким образом, получается т.н. отражение первого порядка (см. рис. 2 «Отражение первого порядка …»). Оно же (отражение первого порядка) есть первое *отрицание* на пути возникновения *языка*. В этом случае отрицается *материя*, а объекты материального мира запечатлеваются в мозгу в виде *образов*/*идей*. «Образы» и «идеи» есть лишь зеркальное отражение

объективного мира в мозгу животных и человека как их высшего представителя.

Отмеченное положение не есть еще собственно *познание*. Оно лишь отражение объективного мира с той точки зрения, где находится субъект отражения.

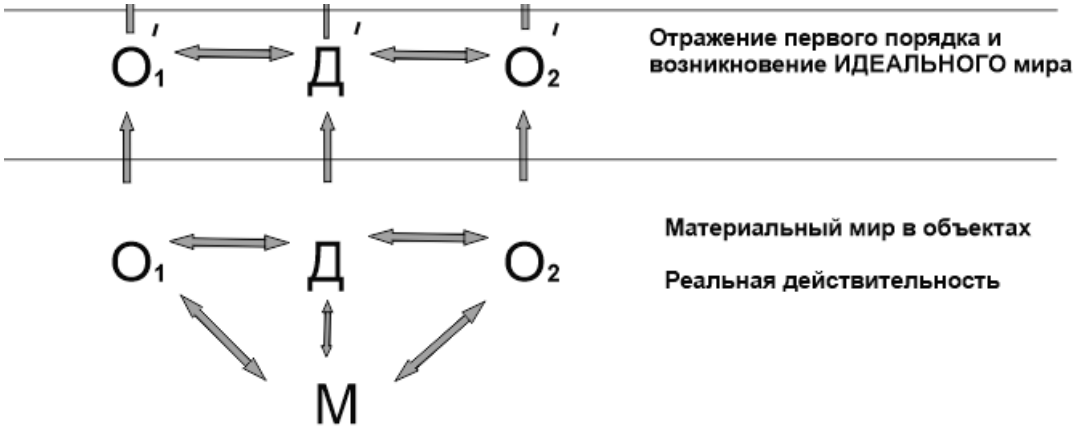

Рисунок 2. Отражение первого порядка

Отражение второго порядка. Говоря далее о животных, будем иметь в виду прежде всего человека, хотя сделанные выводы, по разумению авторов, вполне применимы (в той или иной мере) и к менее организованным животным.

Поскольку для поддержания жизни всякому животному надо обмениваться с окружающей средой, в частности питаться, то возникает потребность выделять из окружающего мира то, что представляет жизненную необходимость для животного, а это требует от него абстрагирования от «ненужных» объектов. Сюда же входит необходимость отличать объекты, представляющие опасность и т.п. Но объекты, очевидно, воспринимаются животным не целиком (в их последней инстанции), а какой-то «одной» своей стороной. Объекты как бы *кажут* или *показывают* себя животному по частям (рис. 2), которые животные фиксируют в мозгу и стремятся удерживать их в памяти, даже если интересующий их объект поменял свое местоположение или исчез из виду, то есть возникает лингвистическое *понятие предиката* (*сказуемого*) и *субъекта* (*подлежащего*) в виде образов (см. рис.3 «Отражение второго порядка …»). Животное начинает «оперировать» *образами* в мозгу или ***со<u>образ</u>ж**ать* (от слова *образ*), целью которых является усилить, то есть как бы увеличить, раздуть до различимого размера те свойства объектов, которые представляют интерес для животного. Животное как бы еще раз само отражает в мозгу уже отраженное. Гиперонимом слова *соображать* является слово *мыслить*. *Мыслить* - значит *думать*. Животное *думает*. Суть этих процессов можно также обозначить словами и словосочетаниями *думать образами, мыслить образами, познавать через образы, знать*. То есть мы получили *мышление* или *разум* и *знание*. Слово *знание* этимологически восходит к слову *знать,*

что означает «делать метку, отличать от остальных или делать помету, что этот предмет видели» [1, с. 145-146].

Таким образом, получаем отражение второго порядка, суть которого проясняет (иллюстрирует) рисунок 3. Животное научается выделять существенное для себя свойство объекта, неважно пища ли это, хищник, соперник или еще что, и превращать его в предикат на образном уровне мышления. На что указывают соответствующие стрелки (рисунок 3).

Выражается все это действиями наблюдаемого объекта или собственными действиями. На этом уровне *отрицается* несущественное (в окружающей действительности материального мира) для животного, а также отрицаются «ненужные» свойства интересующего его объекта. Чисто образное мышление характерно для животного мира.

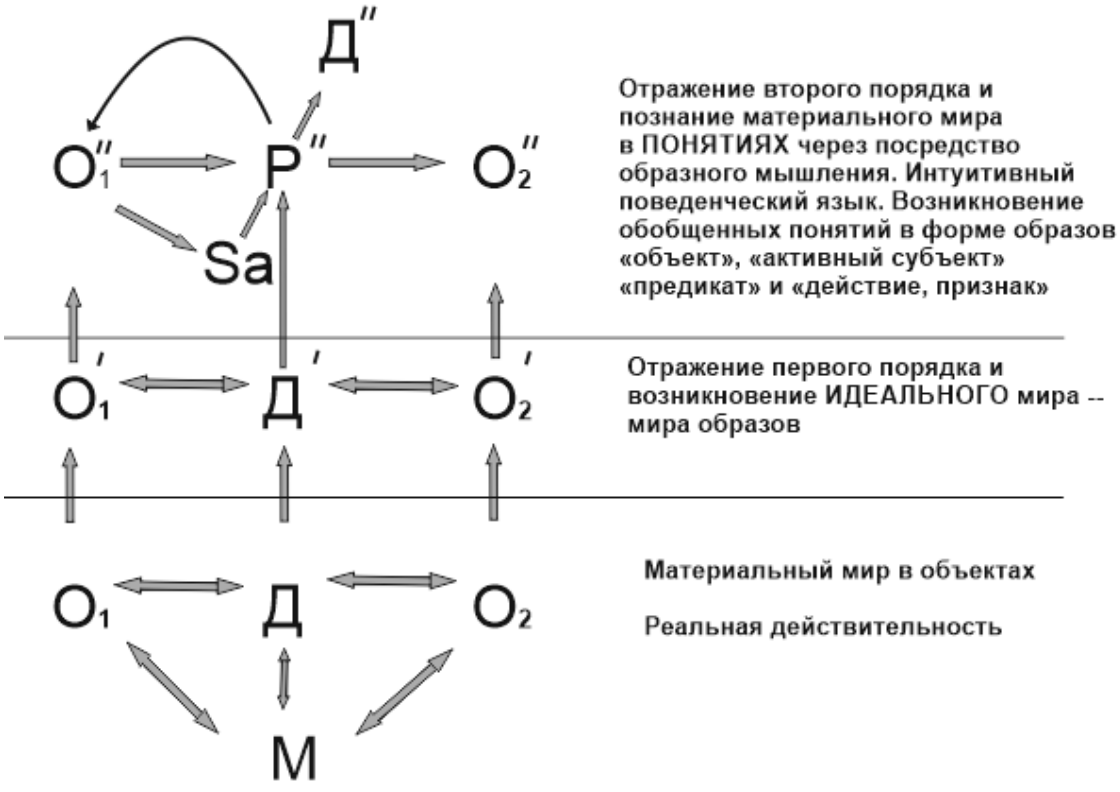

Рисунок 3. Отражение второго порядка

Отражение третьего порядка. Язык. Для дальнейшего изучения материала переходим к понятиям *человек*, *общество* и *язык*. Наиболее точно определение *человека* выводится из работы Ф. Энгельса «Роль труда в процессе превращения обезьяны в человека»: человек – это животное трудящееся, общественное, говорящее и разумное (сознательное) [8, с. 486-498].

Применим к этому определению исторический подход. Люди появились из человекообразных стадных (низшая форма общественной жизни) обезьян, скорее всего, через труд. Стремление выжить и улучшить свое благосостояние требовало от будущих людей вкладывать усилия в

добывание продуктов питания и т.д. и т.п. Это можно было сделать только совместно. Труд в условиях общества требовал объединения усилий его членов. А это, в свою очередь, требовало объединения знаний и опыта каждого индивида, которые суть отражения второго порядка. Объединить знания можно было только через обмен с помощью языка и никак иначе. Возникла система звуков. Надо полагать, что *мама* (в этом звучании) было первым словом вообще, если брать индоевропейские языки. Словами пытались передать мысленные образы «из головы» и посредством жестов передать их свойства. Однако создать точно такой же образ в «чужом» мозгу посредством слов практически невозможно. Поэтому от образного мышления при языковом пути передачи знания люди отказались. Таким образом, объективно, приходим к тому, что человек есть уже социальное животное, постигающее мир в *понятиях*, через *язык* (рис. 3).

А значит, необходимо встает вопрос о третьем порядке отражения объективного мира – через *понятия* и *язык*. На этом уровне отрицается *образ*. Язык реализуется через речь, говорение. Посредством языка передается знание и уничтожается незнание. При передаче знания другому члену общества это знание становится общим или совместным знанием. Так появилось *сознание* – совместное знание. Передать знание об объекте можно только через раскрытие того или иного свойства или качества объекта. Это возможно сделать посредством других слов, произносимых после *имени существительного*, но которыми обозначают сущность познаваемого объекта. Чаще всего это достигается *глаголом*. *Глагол* произошел от слова *голос*, одно из значений которого есть «*зов*», «о*клик*» [1, с. 81, 87]. Иначе говоря, имя существительное должно как бы «прорасти» и показать свою сущность. То есть *слово* переходит в *предложение* через свой *предикат* (Р).

Отражение третьего порядка требует пояснения. Рассмотрим два русских предложения и расставим обозначения над словами согласно тому, что слова означают - материальный объект или его движение (признак):

Ручка^{О} → упала^{Д} → на пол^{О}.

Ручка^{О} → брошена^{Д} → на пол^{О}.

Обратим внимание на один момент. Дело в том, что когда произносится слово *ручка*, то мы его слышим, но еще не слышим слово *упала*/*брошена*, а когда следом произносим слово *упала*/*брошена*, первое слово уже не слышим.

Попытаемся объяснить это явление. Итак, *ручка* есть *слово*, *упала* тоже есть *слово*. С этой точки зрения они равны, ибо *слово* переходит в *слово* и этот переход рождает *предложение*. Но в каждый определенный момент времени из уст одного человека другой человек слышит только одно слово, равно как и произносит человек одно. Поэтому обнаруживаем, что слова отличаются временем произнесения, формой и т.д. *Слова* суть *понятия*. Следовательно, понятие *ручка* переходит/перерастает в понятие *упала*. При этом законы диалектики говорят о том, что понятие *ручка* не просто перешло, а оно стало **основанием** следующего понятия. А слово *упала* (*брошена*) произносится на основании слова *ручка*. Оно положено слову *ручка*

изначально. *Ручка* породило *сказуемое* (Р) и при этом стало *подлежащим* (S). То есть слово *ручка* перешло из *объекта* в *субъект* – теперь слово *ручка* есть *подлежащее*. *Подлежащее* означает «*лежащее под*» (чем-то) [1, с. 311]. Теперь ручка являет себя в своем ином качестве – *упала* (*брошена*) – она себя *кажет, показывает*. Слово *упала* - *брошено* теперь *сказуемое*. Исходя из значения слов *упала* и *брошена* можно выявить характер действия субъекта. В первом случае действие порождается самой *ручкой*, а это значит - *ручка* активна. Таким образом, получаем активный субъект **Sa** и глагол в действительном залоге **Va**. Во втором случае действие направлено на *ручку*. Получаем субъект пассивный **Sp**, а, значит, глагол будет выражен в пассивном залоге **Vp** (см. рис. 4):

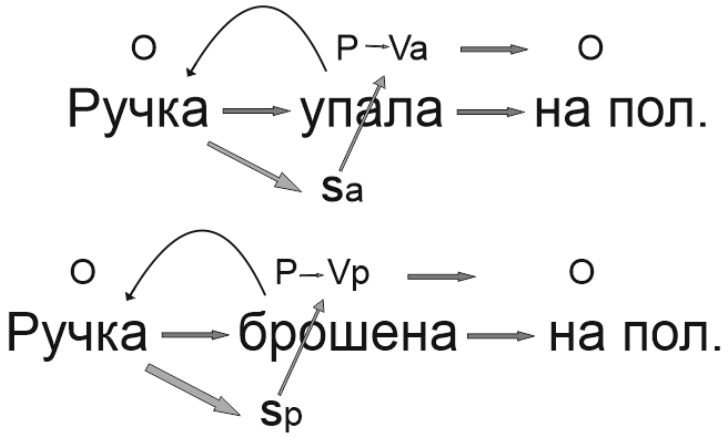

Рисунок 4. Активный и пассивный залоги

Исходя из анализа этих предложений, а также учитывая, что у всякого предложения есть произносящий и слушающий, можем привести формулу грамматического предложения(рис. 5), которую назовем всеобщей формулой познания и языка (ФПЯ):

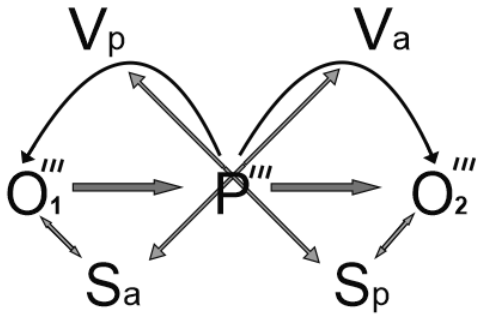

Рисунок 5. Формула познания и языка

Эта формула показывает последний уровень отражения. Общую картину отражения показывает рисунок 6.

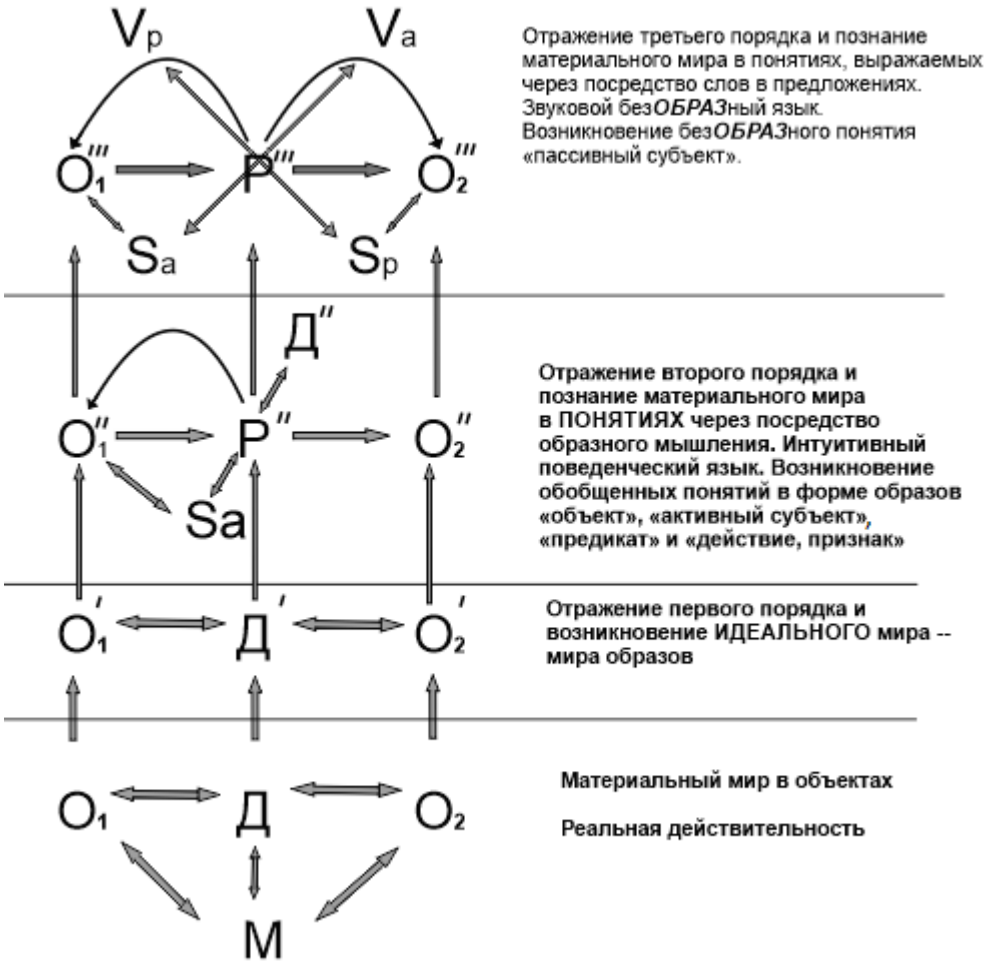

Рисунок 6. Отражение третьего порядка и формула познания и языка

Из формулы (см. рисунок 5) возможно вывести следующие правила:

Sa всегда есть (мыслится) у предиката (действия или признака), но не всегда произносится в предложении. Выражает лицо, число, род.

Sp не всегда может быть (мыслиться) у глагола, но если глагол выражается в форме страдательного залога, то **Sp** всегда произносится в предложении. Выражает лицо, число и род.

Va и **Vp** всегда произносятся в предложении, выражают лицо, число и род подлежащего, а также вид действия, время, залог и наклонение. Если в предложении произносится **Va**, то **Sa** не всегда может быть употреблен. Если в предложении произносится **Vp**, то **Sp** всегда употребляется в предложении.

Исходя из вышеизложенного можно подвести некоторые итоги:

1) *язык* есть движение, а значит

2) субъектом его являются люди, которые

3) с помощью языка передают *знания* друг другу и тем самым

4.) формируют общее знание, то есть *сознание*.

Ну, а поскольку язык имеет место быть только при наличии *знания* и *незнания*, то можно проследить это движение. Если движение идет от знания к незнанию, то человек пользуется повествовательными и побудительными предложениями в зависимости от цели высказывания, а если от незнания к знанию – то вопросительными.

Таким образом**,** язык есть такое орудие общения, с помощью которого люди передают друг другу знание, уничтожая при этом незнание, и тем самым формируют свое общественное сознание.

Теперь рассмотрим пример практического применения ФПЯ (см. рис. 6. Отражение третьего порядка …) при анализе и переводе английского предложения на русский язык. Разберем следующее предложение:

His having gone there is known well. (2.1)

Используя ФПЯ, можно определить, что все элементы данного предложения есть *слова* и с этой точки зрения они все равны (так как все они есть *слова*). Первое слово *his* есть местоимение, обозначающее **объект О** с уже положенным ему **действием Д** *иметь*. Переводится на русский язык словом *его*, а если рассматривать это местоимение с уже положенным ему действием, то перевод будет следующим: *он имеет*. Слово *having* есть глагол, который может быть употреблен в предложении в одной из трех присущих ему функций: смысловой глагол, вспомогательный глагол и эквивалент модального глагола. Слово *gone* есть производное от слова *go*, обозначающее **действие Д**. Слово *there* есть наречие со значением «*туда*», «*то место*», то есть оно означает **Объект О**, *is* есть глагол, который может быть употреблен в предложении в одной из 4 - х функций (глагол-связка, смысловой глагол, вспомогательный глагол, эквивалент модального глагола), *known* есть смысловой глагол в форме $\mathbf{V_3ed}$ и обозначает действие. *Well* есть наречие и означает признак действия. Теперь, согласно ФПЯ, *having gone* есть глагольная форма действительного залога, настоящего совершенного времени (совершенность означает предшествование другому действию в предложении), при этом слово *havin* - вспомогательный глагол, непереводимый на русский язык. То есть это означает, что у действия, выражаемого слов *gone*, есть Sa. Иначе говоря, раз называется действие, то следует обязательно мыслить Sa или определить слово, которое в него перешло. Это слово *his*. Таким образом, можем сформулировать перевод части предложения с одним действием: «он ходил туда», «он сходит туда». Следующая глагольная форма *is known* страдательного залога, настоящего простого времени. Следовательно, в предложении обязательно должен быть употреблен **Sp**. У студентов чаще всего это вызывает трудность с подобного рода предложениями. Но поскольку у любого глагола есть **Sa**, то под словами *is known* поставим знак **Sa** (некто, кто-то). Таким образом можем начать переводить глагол в действительном залоге: «некто знает хорошо, что он ходил туда». **Sa** перешел на дополнение ($\mathbf{O_2}$) «он ходил туда», которое, согласно ФПЯ, переходит в **Sp**. Таким образом, перевод предложения (2.1) получается следующим: «То, что он ходил туда, известно хорошо».

Таким образом, в статье предложена *Всеобщая Формула Познания и Языка* и дан пример практического ее применения при переводе с английского языка на русский.

2. Выведение всеобщей грамматической формулы предложения.

В данной части главы будет кратко показан процесс выведения грамматической формулы предложения из Всеобщей Формулы Познания и Языка (рис. 1, отражение третьего порядка), а также показан пример ее практического использования для выведения типов предложения. Ранее было определено, в чем состоит сущность языка вообще, а также его смысловых единиц – слов и предложений. Они (слова и предложения) есть третий уровень отражения объективной действительности опосредствованные образным мышлением (вторым уровнем отражения объективной действительности). Поскольку было введено первое слово, с которого начинается движение под названием *язык*, то следует найти и первое *предложение*, выражающее суждение. Известно, что с помощью первого слова ребенок всегда обозначает материальный объект. И как определили выше в статье, это слово есть «мама». С помощью слов ребенок называет предметы (живые и неживые). Поскольку все объекты/предметы отличаются друг от друга, то можно утверждать, что имя/название их уже есть суждение о них относительно других объектов. А суждение может выражаться только через предложение. Таким образом, выводится самое первое предложение в языке – *назывное*, часто состоящее из одного слова. Именно из него можно вывести все остальные типы предложений. С помощью назывного предложения «Мама.» ребенок не просто называет того, кто его родил (маму), но также и зовет ее, просит, требует у нее что-то и т.п. Ребенок нередко слово «мама» сопровождает соответствующим просящим жестом, который превращает слово «мама» в обращение, которое в свою очередь переходит в следующее произносимое слово «дай». Не случайно слово «дай» (в русском языке) является вторым по частоте произнесения после первого слова. Таким образом, слово *мама* перешло в назывное предложение «Мама.» и затем перешло в предложение с повелительным значением «Мама, дай.», при этом превращаясь в обращение.

С помощью органов чувств человек непосредственно познает окружающий его мир и формирует свое *знание* о нем, то есть формирует то, чем он будет пользоваться в своей жизни. Поскольку человек есть существо общественное, это значит, что у него выработалась потребность в помощи других членов общества, а точнее, в их знаниях и умениях. *Язык* есть средство получения этих *знаний*, но уже без непосредственного отражения изучаемого объекта. С помощью *языка* передают *их* (*знания*) другому человеку, и, таким образом, формируем *сознание* (совместное знание). Надо полагать, что первое *слово* (мама) уже есть начало формирования *сознания*, ибо появляется нечто общее в памяти, чем люди начинают пользоваться. Таким образом, можно утверждать, что без *языка* не существует *сознания*.

Знание о чём-то можно передать только тому, у кого это знание об этом чём-то отсутствует или оно неполное. Таким образом, мы получили движение от *знания* к *незнанию*. Такое движение реализуется с помощью повествовательных предложений. Надо отметить, что в слове

«повествовательный» элемент «вест(ь)» является ключевым и обозначает «сообщение, знание» [1, с. 55].

Существует и обратное движение – от *незнания* к *знанию*. Такое движение реализуется вопросительными предложениями, которые являются производными от повествовательных. Они предназначены для устранения своего незнания в чем-то, в чем есть потребность. Это подтверждается тем, что слово *просить*, от которого произошли слова *вопрос* и *вопросительный*, означает «требовать, умолять», то есть это есть требование устранить незнание в чем-то, что для спрашивающего может быть или может оказаться жизненной необходимостью.

Таким образом, получается следующая цепочка: слово – назывное предложение – повелительное предложение – повествовательное предложение – вопросительное предложение. Получается весь набор простых предложений, отражающих суть нашего движения под названием *язык*.

Как всякое движение (или совокупность действий) предложение всегда когда-то начинается и обязательно заканчивается. Это *закон*. Все подчиняется этому *закону,* и этот *закон* еще и управляет. Другими словами, возникают *правила* произнесения (написания) предложения. Иначе говоря, подчиняясь законам речи, необходимо следовать правилам произнесения слов в предложении. Невозможно произвольно произносить слова. В противном случае человека просто не будут понимать. Таким образом, *правила* эти становятся едиными для той или иной группы людей, того или иного общества. Постепенно сложилась ситуация, когда люди стали изначально подчинять свою речь этим правилам. То есть сначала необходимо определить правила предложения, прежде чем произносить его.

Таким образом, потребность в чем-то заставляет одного индивида, не имеющего возможности самостоятельно удовлетворить потребность, обращаться к другому (обращение, что соответствует назывному предложению) за помощью. Младенец, например, не в состоянии сам себя накормить и т.п. Поэтому при дискомфорте (что, по сути, есть движение живого организма к его смерти) он называет тот объект, от которого зависит его благосостояние (то есть движение к жизни), иначе говоря, когда он зовет маму, произнося слово *мама*, что может означать «Иди сюда», «Дай молока» и т.п и т.д. То есть потребность определяет цель высказывания. Соответственно цели подбирается тип предложения (побудительное, повествовательное, вопросительное).

Итак, получены первые элементы грамматической формулы простого предложения. Обозначим *предложение* латинским **Pr** (лат. propositio). Также будем указывать тип предложения: повелительное **Imp** или **Пов** (лат. ***imp***erativus - ***пов***елительный), повествовательное **Nr** или **Пв** (лат. ***n***a***r***ratorius - ***п***о***в***ествовательный), вопросительное **Int** или **?** (лат. ***int***errogativus - вопросительный).

Согласно ФПЯ [9, с. 336-352] (см. рис. 5) в предложении называемый объект в момент полагания ему предиката превращается в субъект.

Обозначим его знаком **S** (лат. ***s***ubjectum - субъект). В статье [9] было показано, что в предложении рассматриваемый объект может превратиться или в активное подлежащее, или в пассивное (страдательное). Получаем следующее:

Pr Imp Sa– повелительное предложение с активным подлежащим;

Pr Nr Sa/p – повествовательное предложение с активным или пассивным подлежащим;

Pr Int Sa/p – вопросительное предложение с активным или пассивным подлежащим.

Формат главы не позволяет отразить здесь такие моменты, как придаточное предложение, безличное и т.п., а у подлежащего – лицо, число, род. Поэтому такие подробности опустим.

Обязательным элементом предиката является глагол (правда, иногда он не произносится в некоторых языках в настоящем времени). Его обозначим буквой **V** (лат. ***v***erbum - глагол). Из ФПЯ видно, что действие или порождается объектом, который при этом превращается в активный субъект **Sa**, или принимается объектом, который, при полагании ему действия, превращается в страдательный/пассивный субъект **Sp**. Таким образом, можно вывести два залога глагола – действительный **Va** и страдательный/пассивный **Vp,** в выводимой формуле конкретного предложения следует указывать залог глагола. Называемое действие, выражаемое глаголом-сказуемым, может отрицаться говорящим или подтверждаться. Это рассматривается как отрицательное предложение или как положительное. Отрицание выражается через глагол-сказуемое. Положительное обозначим знаком «+», отрицательное – знаком «–». Также с помощью глагола обозначаем время, вид и наклонение. Наклонение обозначим знаком «m» (от лат. ***m***odus), время – знаком «**t**» (от лат. ***t***empus – время), вид действия – знаком «**a**» (от лат. ***a***spectum - вид). Таким образом, получаем подлежащее и соответствующее ему положительное или отрицательное сказуемое, которые могут быть произнесены в любом из типов предложений:

Sa/p +/– Vamta – подлежащее и глагол-сказуемое.

Положим данную систему (S+/–V) на тип предложения и таким образом получим полную грамматическую формулу предложения (ФП) с указанием на тип предложения. Например, грамматическая формула для вопросительного предложения может выглядеть следующим образом:

Pr ? Sa/p + Va/pmta (предложение вопросительное, с активным /пассивным подлежащим, положительным глаголом в действительном /страдательном залоге, в нужном наклонении, времени и виде).

Таким образом, получается грамматическая формула предложения, выведенная из ВФПЯ. Ниже будет рассмотрен пример практического применения при обучении переводу с русского на английский.

Исходя из проведенных построений, можно определить алгоритм действий при таком переводе. Очевидно, что сначала определяется тип предложения (повествовательное, вопросительное, побудительное), статус предложения (безличное, придаточное и т.п.), затем определяются

подлежащее и полагаемый ему глагол-сказуемое, для которого требуется определить залог, наклонение, время и действие.

Следует отметить, что *язык* имеет две стороны своего бытия – активную сторону и пассивную. Активная сторона *языка* подразумевает движение от содержания к форме, это есть движение человека, произносящего предложение. Пассивная сторона языка есть движение от формы к содержанию, осуществляемое человеком, воспринимающим предложение. Значит, есть правила речи (грамматики), которым подчиняются произносящий предложение и воспринимающий это предложение. Эти правила отличаются друг от друга. Таким образом, можно говорить о двух сторонах грамматики языка вообще (грамматических правилах и алгоритмах их составления) – активной и пассивной [10]. Поскольку в работе рассматриваются два языка и переводы с одного из них на другой, то логично предположить, что для русскоязычного человека грамматика, используемая для перевода с родного на иностранный язык, будет считаться активной.

3. Активная и пассивная грамматика естественного языка в философском, лингвистическом и психологическом аспектах

В этой части главы будут уточнены понятия «активная и пассивная грамматика», рассматриваемые с позиций работ [10, 11,12, 13, 14, 15 16]. Термин «активная и пассивная грамматика» впервые применил известный советский языковед академик Л.В. Щерба. Он пришел к выводу, что можно описать язык в соответствии с тем, как человек говорит, а можно - в соответствии с тем, как он воспринимает чужую речь, и эти описания будут разными. Ведь в одном случае человек идет от какого-то содержания или смысла, который он хочет выразить, и для этого подбирает нужные языковые средства. А в другом случае у него эти средства уже есть (сформированы) и проблема в том, чтобы понять, что они означают (то есть какое знание они несут) и для чего служат. Активная грамматика, следовательно, строится по принципу "от содержания к форме", а пассивная - "от формы к содержанию"[10].

К сожалению, до недавнего времени не было создано ни активной, ни пассивной грамматик ни для одного языка. А это является одной из главных причин того, почему процент научившихся владеть речью на иностранном языке весьма мал и составляет всего чуть больше 1 % всех изучавших его в образовательных цепочках "школа - среднее учебное специальное заведение - вуз" и "школа - вуз". Это показал опрос студентов-первокурсников (103 человека), окончивших среднюю школу, и косвенный опрос их родителей (206 человек), окончивших вузы и средние специальные учебные заведения. Только три человека из 309 опрошенных заявили, что они владеют иноязычной речью свободно. Все они оказались родителями студентов. Двое из них закончили факультет иностранных языков, а один научился самостоятельно и то только потому, что ему подолгу приходилось жить и работать за границей. В то же время есть люди, которые могут говорить и понимать на нескольких языках и при этом никогда не изучали их в учебном

заведении. Причина успеха последних, на наш взгляд, кроется в естественности овладения навыками говорения и восприятия речи. Некоторые причины неуспеха первых уже указывались в более ранних публикациях [13].

Анализ наблюдений показал, что изучается язык в описательной форме, причем он изучается в отрыве от человека как биологического существа, которое является непосредственным носителем речемыслительных действий и сознания. Именно поэтому практически почти ни одно упражнение на отработку речевых навыков не соответствует реальной действительности по способу их выполнения. Например, задание подставить нужное слово в уже написанное предложение с пропущенным местом; получается в этом случае, что предложение произносит один человек (автор учебника), а изучающий иностранный язык лишь вставляет в чужие фразы отдельные слова или словосочетания, уже данные автором учебника. Но речь есть результат и отражение речемыслительной деятельности человека, что подразумевает соответствующую методику обучения владению речью на иностранном языке.

Дескриптивная грамматика описывает лишь письменный («мертвый») язык, оторванный от человека, как его носителя. Об этом можно судить по следующим заданиям: выберите правильный вариант ответа, подставьте в пропущенное место нужное слово, поставьте слово в нужную форму, поставьте предложения в вопросительную и отрицательную формы, определите форму слова и т.п. Научить говорить с использованием такой грамматики по правилам иностранного языка и воспринимать иноязычную речь крайне сложно, почти невозможно (как было аргументировано выше). А живая речь порождается и воспринимается совершенно по другим правилам. Прежде чем рассматривать данную проблему далее, рассмотрим понятия "язык", "речь" и "предложение", а также найдем причину, в чем кроются недостатки в понимании данных терминов. Не существует точного и общепринятого определения для всех этих понятий.

Философское понятие "язык" подразумевает, что это "система знаков, служащая средством человеческого общения, мышления и выражения" [2, с. 816]. Многие ученые по-разному определяли "язык": Г. Штейнталь, А.А. Потебня, И.А. Бодуэн де Куртенэ - как процесс, Г. Пауль, К. Бругман - лишь как систему "психических образов", Ф. де Соссюр - как способность, функцию индивида, Л.В. Щерба – как нечто единое и общеобязательное для всех членов данной общественной группы, гештальтисты - как крайне сложную психофизиологическую функцию, бихевиористы (Дж. Уотсон, Э. Торндайк, Л. Блумфилд) - как простую количественную прибавку к другим стимулам [17, с.26-33]. И.П. Сусовым язык "понимается как динамическая система, обеспечивающая многоступенчатое кодирование значения в звуках и обратное движение от звуков к значениям" [18]. Последнее определение подводит к тому, как необходимо разрабатывать активную и пассивную грамматику, однако только оно (определение) составлено по отношению к звуковому языку.

"Речь" в философском словаре определяется как "речевая деятельность, общение, опосредствованное языком, один из видов коммуникативной деятельности" [2, с. 582]. Специалистами речь определяется как исторически сложившаяся форма общения людей посредством языковых конструкций, создаваемых на основе определённых правил. Процесс речи предполагает, с одной стороны, формирование и формулирование мыслей языковыми (речевыми) средствами, а с другой восприятие языковых конструкций и их понимание. Таким образом, речь представляет собой психолингвистический процесс, форму существования человеческого языка [19]. Добавим, речь есть необходимое условие существования языка.

Отделим теперь друг от друга понятия *язык* и *речь*. В результате обнаружим, что во время говорения человек двигает органами речи, поэтому речь мы можем определить в этом случае как движение, но движение, которое присуще только человеку. Если будем прислушиваться к речи, то обнаружим в ней закономерности, которые можем трактовать как законы языка, и эти законы (правила) являются едиными для определенной общности людей. Таким образом, приходим к выводу, что речью управляют законы этого движения – законы речи или языка. Возникает понятие *грамматические правила языка*. Теперь посмотрим, как соотносятся друг с другом язык, правила языка и речь.

Рассмотрим речь с точки зрения говорящего. Закономерности языка заставляют человека выстраивать свою речь из временны*х* отрезков движения своих органов речи и голосовых связок. Эти отрезки принято называть *словами*. Объединяя эти отрезки речи (*слова*) интонационно в значимые группы слов, мы образуем *предложения*. Человек не может произносить предложение, не следуя уже готовым правилам, в противном случае это будет набор звуков. Значит, чтобы говорить, необходимо сначала иметь этот конкретный для каждого предложения набор (систему) правил. Поскольку движение идет от содержания и цели высказывания, то подбирается тип предложения (повествование, вопрос и т.п.) и соответственно этому формируется индивидуальный набор правил [11, 12, 16]. Через речь человек воздействует на сознание слушателя, заставляя его делать что-то (пойти, сказать, дать информацию о чем-то и т.п.). Говорящий воздействует на объект, значит, он активен. Иначе говоря, говорящий человек использует *активную речь*.

Теперь рассмотрим речь с точки зрения слушающего. Логично предположить, что слушающий не может знать, с какого слова начнется предложение и по каким правилам предложение будут произносить. Эту задачу он должен решать по мере восприятия слов в предложении. Алгоритм мыслительных действий по анализу предложений задается последовательностью произносимых слов. Для восприятия слушающий вынужден держать все правила языка наготове и последовательно снимать те или иные задачи. Полное совпадение грамматической формулы предложения (индивидуальное правило предложения), составленной слушающим, с такой же грамматической формулой предложения, но составленной говорящим,

будет говорить о понимании. Говорящий зашифровывает, слушающий расшифровывает. Говорящий использует ключ (активную грамматику), чтобы зашифровать в слова смысл, слушающий использует ключ (пассивную грамматику) чтобы расшифровать смысл предложения. Чтобы понять смысл предложения, слушающему приходится адекватно выстраивать услышанное предложение мысленно, в собственной голове. Этот процесс и есть *пассивная речь*.

Человек, как известно, наделен природой способностью к речевой и речемыслительной деятельности. Это подсказывает, что человека нужно обучать речевой деятельности, а не изучать язык в его описательной форме. К сожалению, на протяжении многих лет практики грамматика составлялась и составляется доныне для языка, предварительно отчужденного от речемыслительной деятельности человека, и изучалась и изучается как нечто самостоятельное. Достаточно посмотреть, например, образцы порядка синтаксического разбора русских словосочетаний и предложений при изучении русского языка [20, с. 240-246]. В реальности обычный человек анализирует элементы воспринимаемого предложения последовательно, по мере их восприятия [13, 14, 21].

4. Практическое применение активной грамматики при обучении переводу с русского языка на английский

В 2015 году опубликовано учебное пособие «Активная грамматика английского языка» [11, 12]. Пособие, прежде всего, предназначено для аспирантов и ученых математических специальностей, которым необходимо писать свои статьи и доклады на английском языке, что для русскоязычного человека означает перевод с родного языка на иностранный язык. Рассмотрим в контексте статьи некоторые принципы, положенные в основу данного пособия, в частности то, что непосредственно касается грамматической формулы предложения. В пособии даны общие сведения об английском предложении, определитель нужной схемы и общий вид схем предложений (см. рис. 7), с помощью которых можно перевести русские предложения на английский язык. Для работы с русским предложением разработан специальный алгоритм подбора правил, который позволяет пошагово самостоятельно составить индивидуальную формулу предложения (ФП) и определить нужную схему.

Пояснения: ПрстГлПрПр – простое, главное, придаточное предложение (выбирается нужное); ПобПр – побудительное предложение;

ПростПовеств., ПрПр или ?п – простое предложение, повествовательное, придаточное или вопрос к подлежащему;

? – вопросительное предложение с обратной последовательностью главых членов предложения;

Рисунок 7. Схемы предложений

V/be – глагол, причем глагол be из-за его функциональных свойств противопоставляется всем другим глаголам;

НП, ПП и др. –обозначения времени и действия глагола (Н_, П_, Б_, Бп_ означают соответственно настоящее, прошедшее, будущее и будущее в прошедшем времена; _П, _Д, _С, _Д означают простой, длительный, совершенный и длительно-совершенный виды действия);

Стр. – страдательный залог;

ИЧС/ДП – именная часть сказуемого, дополнение;

Об – обстоятельство;

W/C – вопросительное слово, относительное местоимение или союз;

Пд/W?п – подлежащее, а также подлежащее, выражаемое вопросительным словом или словосочетанием в вопросе к подлежащему;

ВГ/М – вспомогательный или модальный глагол;

1. Исходя из цели высказывания, определите **тип** предложения по плану:

Таблица 1

<table>
<tr><td colspan="5">Простое, сложносочиненное, сложноподчиненное предложение</td></tr>
<tr><td colspan="3">Простое, главное</td><td colspan="2">Придаточное предложение</td></tr>
<tr><td rowspan="3">ПобПр [1]: 1.«(не) делайте^V», «будьте^V»; 2. «Пусть он (с)делает», «давайте я (мы) …» = «Разрешите^V (let) ему (мне, нам) (с)делать». Определите глагол, «+» или «–»,затем переходите к п. 5.</td><td rowspan="3">?п[24] *</td><td>Пов[24]; ?[35]</td><td>Пов.[24], ?[24]косв</td><td>?п[24]*</td></tr>
<tr><td colspan="2">Безл / ИмБезл / Сущ</td><td rowspan="2">Переходите к п. 2</td></tr>
<tr><td>Нет</td><td>Да</td></tr>
<tr><td colspan="2">Переходите к п. 2</td><td>Впишите в ФП Безл., ИмБезл или Сущ. и переходите к п. 2</td></tr>
<tr><td colspan="5">*(<u>Кто (из них)</u> делает…? <u>Чей/какой друг</u> (<u>Сколько человек)</u> делает…? <u>Что</u> лежит…?)</td></tr>
<tr><td colspan="5">Любое предложение может быть восклицательным.
Или пишите / произносите слова по следующей схеме:
«Какой это дом!»= «Что это за дом!» – What a house it is!; «Как же быстро он сделал это!» – How quickly he did it!</td></tr>
</table>

«+» или «–» – положительное или отрицательное предложение.

Приведем сам алгоритм подбора грамматических правил.

Чтобы письменно перевести предложение на английский язык, сначала составьте **формулу предложения** (**ФП**) по алгоритму.

2. Определите с помощью таблицы 2 **глагол-сказуемое, модальные слова.**

Таблица 2

<table>
<tr><td colspan="3">Слова: 1. «Мочь, уметь, можно, нельзя»; 2. «Надо, должен, следует, нельзя приходится [ЭМ have to, при этом дополнение (кому?) есть логическое подлежащее]».</td></tr>
<tr><td>«НЕТ»</td><td colspan="2">«ДА»</td></tr>
<tr><td>Определите глагол-сказуемое* или страдательное причастие** в положительной (+ , делает, сделано) или отрицательной («–» не делает, не сделано) форме, а также его залог*. (Если предложение будет БезлПр или Сущ., то его определяйте с помощью этих правил и впишите в ФП) Запишите V или V₃ed* над глаголом в личной форме или над страдательным причастием.
Глаголы be (быть, находиться, заключаться в, являться), have (иметь), going to (собираться) и некоторые другие необходимо вписывать в ФП</td><td>М⁴⁵ ~~то~~ инф*
Все случаи модальности.
Определите время и действие и наклонение (см. п. 3 алгоритма), исходя из этого определите вид инфинитива и впишите его над глаголом, при этом must, need, should, ought to в прош. времени замените на ЭМ «have to V/be V₃ed*». Напишите над словом знак «М» и впишите «М⁴⁵ ~~то~~ » в ФП (то есть can, could, may, might, must, need, ought to, should, shall) со знаком «–» или «+»; а над инфинитивом напишите знак V или V₃ed*</td><td>ЭМ to + простой инф.*
Конкретный случай, вынужденность, подразумевается наличие обстоятельств и причины, ~~просьба,~~ ~~совет,~~ ~~предложение.~~ ~~разрешение~~
Напишите над словом «ЭМ», впишите его в ФП со знаком «+» или «–»; над инфинитивом напишите знак V или V₃ed* и впишите его в ФП:
«be (able to V/be V₃ed*)^{ИЧС}»,
«be (allowed to V / be V₃ed*)^{ИЧС}»,
«be (to V / be V₃ed*)^{ИЧС}»,
«have to V / be V₃ed*».
То, что заключено в скобках, нужно будет произносить на ИЧС</td></tr>
<tr><td colspan="3">* В русском предложении в настоящем часто опускаются глаголы со значениями "есть, находится". Попробуйте произнести предложение в прошедшем времени, если произносится слово "был", то впишите в ФП глагол be. В предложении типа «У меня (у друга) есть книга» глаголом является слово "иметь" – have [Он (мой друг) имеет книгу].</td></tr>
<tr><td colspan="3">**Страдательному залогу соответствуют русские предложения одной из следующих структур: 1. «Это (должно/может быть) сделано», 2. «Дом строится», «Дом должен/ может строиться». 3. «Это кому-то (с)делали-(ют)», «Это кому-то должны/могут (с)делать». В этом случае над глаголом или страдательным причастием напишите знак V₃ ed. Если предложение соответствует типу 3, то одно из дополнений будет подлежащим (подчеркните его как Дп и затем как Пд), при этом поменяйте «?³⁵» на «?п²⁴», в случае если вопросительное слово (сочетание) является дополнением-подлежащим</td></tr>
</table>

3. Определите с помощью таблицы 3 наклонение, время и вид глагола с помощью таблицы соответствия видовременных форм глагола.

Таблица 3

Русский глагол	Английский глагол в форме	
П (с)дела***л***, бы***л***; **Н** делает, "есть", находится; **Б*** сделает, будет делать, будет; собираться, намереваться	**ПП** или/и **НС**[45] **НП** или/и **НД**[45] **БП**[45]***** или/и **НД**[45]***** ***be*** в **длит**. нельзя	**ПД**[45], **ПС**[45], **ПДС**[45] **НС**[45], **НДС**[45] **БД**[45], **БС**[45], **БДС**[45]
*** НД** может выражать запланированное действие в будущем. Если глагол в будущей форме, то попробуйте произнести предложение со словом «*собираться, намереваться*»: "*Я напишу это*" – "*я* ***собираюсь/намереваюсь*** *написать это*". Если это возможно, то можно брать второй вариант со словом "*собираться/намереваться*". Введите в **ФП** запись … g**o*ing to V / be V3ed.*** "*Стол* ***собирается/намеревается*** *стоять в углу*" – вариант невозможен, значит, **НД** не подходит. Глаголы, обозначающие движение, можно переводить сразу в форме **НД**		

4. Уточните схему: схемы « 2», « 3» предназначены только для одного глагола ***be*.**

5. Прочитайте текст над схемами (*произносите* ***артикль*** *или местоимение перед каждым существительным*) и держите эту фразу в голове, пока не построите предложение. Нельзя применять правило "**артикль**", если при существительном произносятся местоимения в роли определения: **мой -** *my* и другие. Произносите/пишите предложение строго по схеме. Соединённые чертой члены предложения вы должны обязательно произносить или подразумевать, остальные – по смыслу. Вопрос к подлежащему начинайте произносить с **Пд/W?п.**

Теперь приведем пример рассуждения с краткой записью формулы по приведенному алгоритму:

«**Вчера профессор сказал, что конференция начнется сегодня».**

1. Предложение является сложноподчиненным, состоящим из двух предложений. Первое предложение («**Вчера профессор сказал**») – главное, второе («**что конференция начнется сегодня**») – придаточное.

Главное предложение является простым, повествовательным. Правила «Безличные предложения» и «Существование чего-либо» применить нельзя в данном случае. Подлежащее «профессор» в предложении активное. Записываем:

Вчера профессор сказал, что конференция начнется сегодня.

ПрГл Пов[24]

2. Модальных глаголов в главном предложении нет. Глагол «сказал» положительный и произносится в форме действительного залога. Дополняем формулу:

Вчера профессор сказал[V], что конференция начнется сегодня.

ПрГл Пов[24] + Дз

3. Уточняем наклонение. Глагол «сказал» стоит в форме изъявительного наклонения, настоящего времени и простого вида действия. Дополняем формулу:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП

4. Уточняем схему. По совпадению Пов + ПП определяем, что это схема 2 [по схеме 4 + (положительная) нельзя произнести / записать слова в предложении, так как она (схема) не предназначена для выражения прошедшего времени с помощью глагола в простой форме]:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП 2

5. Разбираем второй элемент предложения «**что конференция начнется сегодня**».

Предложение придаточное дополнительное, и, следовательно, в нем будем применять правило согласования времен. Предложение является повествовательным, с активным подлежащим «конференция», и это означает, что глагол «начнется» будет в форме действительного залога. Предложение вводится союзом **that**:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП 2

ПрДп Соглас.

6. Модального глагола в предложении нет, а смысловой глагол «начнется» является положительным в форме действительного залога:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП 2

ПрДп Соглас. Пов^{24} + Дз

7. Уточняем наклонение, время и действие смыслового глагола. Глагол «начнется» имеет форму изъявительного наклонения, будущего времени и простого вида действия – исходное время БП^{45}. Но по правилу согласования времен (на основании записи **ПрДп Соглас.**) изменяем время БП^{45} на БпП^{45}:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП 2

ПрДп Соглас. Пов^{24} + Дз Из ~~БП~~^{45} → БпП^{45}

8. Уточняем схему. Пов^{24} и БпП^{45} имеют в верхнем индексе общую цифру «4», что и есть номер схемы:

Вчера профессор сказал^{V}, что конференция начнется сегодня.

ПрГл Пов^{24} + Дз Из ПП 2

ПрДп Соглас. Пов^{24} + Дз Из ~~БП~~^{45} → БпП^{45} 4

Или другой вариант записи:

ПрГл Пов^{24} 2 + Дз Из ПП

ПрДп Соглас. Пов^{24} 4 + Дз Из БП^{45}

Расшифровывается краткая запись (уже применительно к английскому предложению) так:

Предложение сложноподчиненное. Первое предложение – главное, простое, повествовательное с порядком слов по схеме 2, положительное,

глагол будет произноситься/записываться в форме действительного залога, изъявительного наклонения, настоящего времени и будет обозначать простой вид действия.

Второе предложение придаточно-дополнительное (= изъяснительное), а поскольку в главном предложении глагол в форме прошедшего времени, то будет применяться правило согласования времен. Предложение повествовательное с порядком слов по схеме 4, положительное, глагол будет произноситься/записываться в форме действительного залога, изъявительного наклонения, будущего времени и будет обозначать простой вид действия.

9. Перевод:

Yesterday the professor said that the conference would begin today.

Таким образом, в этой части главы приведен алгоритм вывода грамматической формулы каждого конкретного предложения в режиме перевода с русского языка на английский.

На основе данного исследования разрабатывается концепция электронного учебника для практических занятий по переводу технических текстов на английский язык

В последние десятилетия английский язык занял особое положение в научном мире. К современным специалистам, аспирантам, молодым ученым предъявляются требования переводить на английский язык свои тезисы докладов, научные статьи и пособия. Многие научные конференции имеют статус международных, на которых используются два рабочих языка, в нашем случае это русский и английский. Это обусловило разработку электронных обучающих средств, способствующих более эффективному овладению техникой перевода с русского языка на английский [22].

Необходимо выделить следующие существенные моменты:

1) аспиранту, молодому ученому, переводящему текст с родного (русского) языка на английский, приходится иметь дело с двумя языками;

2) почти всегда студенты технических специальностей, равно как аспиранты и молодые ученые, не являются профессиональными переводчиками.

Первый момент обусловливает необходимость практического владения грамматическими правилами русского и английского языков, а также умения правильно находить грамматические соответствия в процессе перевода. В противном случае это приводит к ошибкам или вносит двусмысленность в текст. Второй момент требует учитывать отсутствие у аспирантов и молодых ученых достаточного опыта переводческой деятельности. Для них характерен небольшой словарный запас, что вынуждает их часто обращаться к словарям и тратить на это время. Они часто ошибочно употребляют английские слова, соответствующие одному и тому же русскому термину, например, *variety* вместо *manifold*.

5. Разработка пассивной грамматики и практическое применение ее при обучении студентов с использованием электронных средств и вопросы алгоритмизации машинного перевода текстов

Цель этой части заключается в том, чтобы ознакомить заинтересованное научное сообщество с созданием электронного учебного пособия по активной грамматике английского языка, помогающего развить умение грамотно передавать мысли, возникающие на русском языке, средствами английского языка.

Электронное обучающее средство по активной грамматике английского языка должно:

1) предоставить работающим с данным пособием возможность быстро находить справочные материалы по тем или иным грамматическим явлениям русского и английского языков именно в момент процесса перевода;

2) содержать достаточное количество примеров перевода с русского языка на английский;

3) содержать устойчивые речевые клише, используемые в научных текстах;

4) учитывать техническую направленность студентов, аспирантов и молодых ученых, особенно в области математики, физики и информатики;

5) иметь простой и понятный интерфейс.

Для решения первой задачи авторы воспользовались моделью *активной грамматики английского языка* (грамматики, используемой при построении предложений, в противовес *пассивной*, используемой при анализе чужой речи), предложенной в [23]. В этой работе авторы приводят последовательность действий, которую следует выполнять для решения задачи перевода предложения на английский язык.

Согласно этой модели, английское предложение строится в соответствии с одной из пяти обобщённых схем: одна (№ 1) для побудительных (положительных или отрицательных) предложений), две (№ 2 и № 4) для предложений с прямой последовательностью произнесения/написания главных членов, две (№ 3 и № 5) – для предложений с обратной последовательностью главных членов (см. рис. 7).

Алгоритм перевода предложения на английский язык включает в себя 5 шагов (см. таблицы 1, 2, 3).

В работе [21] приводится пример построения предложения по данному алгоритму и комментируются нюансы применения конкретных схем и правил; похожий пример включён также и в разработанное электронное пособие.

Первая часть разработанного пособия включает следующие разделы:

- общие сведения об английском предложении;
- краткое описание схем предложений (согласно классификации в [21]);
- обобщённый определитель нужной схемы;
- общий вид каждой из схем;
- время и действие глагола, вспомогательные глаголы и согласование имён;

- алгоритм построения формулы предложения;
- детальный разбор примера построения формулы предложения;
- последовательность произнесения главных членов предложения (прямая и обратная) и особые случаи;
- главные члены предложения;
- второстепенные члены предложения;
- глаголы (роли в предложении и порядок употребления);
- модальные глаголы и их эквиваленты;
- передача времён английским глаголом в предложении;
- согласование времён в придаточных предложениях;
- виды действия в изъявительном наклонении;
- сослагательное и повелительное наклонения;
- страдательный залог;
- неличные формы глагола;
- инфинитив;
- герундий;
- причастия I и II;
- безличные и именные безличные предложения;
- оборот there be;
- числительные;
- союзы и союзные слова;
- придаточные предложения;
- правила употребления артикля.

Вторая часть электронного пособия представляет собой обширный набор типовых оборотов речи, характерных для научно-технических текстов на английском языке, готовый для использования при переводе научно-технических текстов и докладов.

Все примеры разбиты на следующие категории:
- цели и задачи исследования;
- формулировка проблем;
- объект исследования;
- декомпозиция целей и задач;
- методы и подходы;
- побуждение к анализу и уровни анализа;
- характеристика проблем и вопросов;
- описание степени изученности задач;
- обзор литературы;
- изложение существующих точек зрения;
- высказывания о теории;
- выдвижение гипотез;
- изложение общепринятых положений;
- презентация источников;
- изложение и характеристика имеющихся материалов;
- классификация материала;
- ссылки на источники;

- начало процесса информирования;
- продолжение процесса информирования;
- выражение связи с последующим изложением;
- выражение связи с предшествующим изложением;
- возвращение к сказанному ранее;
- пауза в процессе изложения;
- уточнения и разъяснения;
- внесение дополнений;
- активизация внимания;
- изложение собственной точки зрения;
- иллюстрация теоретических положений;
- ссылка на авторитетные источники;
- высказывание предположений;
- объяснение;
- аргументация;
- верификация;
- выражение уверенности или сомнения;
- воссоздание полной картины;
- изложение фактов;
- установление связей и различий между имеющимися фактами;
- установление причин и следствий;
- выделение характерных факторов, черт, явлений;
- способы обозначения исторического времени;
- заключение, выводы, подведение итогов;
- выражение благодарности.

Третья часть разработанного электронного ресурса представляет собой справочный материал, полезный при переводе математических текстов.

Важной особенностью подобранного материала является наличие транскрипций для каждого из математических знаков, сокращений, а также греческих букв.

Справочный материал электронного пособия разбит на следующие категории:

- греческий алфавит с транскрипцией;
- таблица математических знаков и функций;
- дополнительный набор терминов и понятий, связанных с операциями сложения, вычитания, умножения, деления, дробей, возведения в степень;
- дроби;
- отношения и пропорции;
- математические уравнения;
- примеры чтения математических формул;
- таблица сокращений.

Созданное электронное пособие является исправленным и дополненным вариантом выпущенного ранее учебного пособия [11, 12] и позволяет обучающимся пользоваться удобствами, обеспечиваемыми современными

ЭВМ при работе с текстом (контекстный поиск, быстрая навигация по оглавлению, возможность копирования, наличие гиперссылок).

Электронное пособие подготовлено в формате Compiled HTML (CHM) [11, 12], распространяется в виде одного файла обычным копированием и ввиду широкого распространения формата открывается на любом современном компьютере под управлением ОС Windows и не требует для работы подключения к сети Интернет.

На рисунке 8 показан внешний вид оглавления электронного пособия.

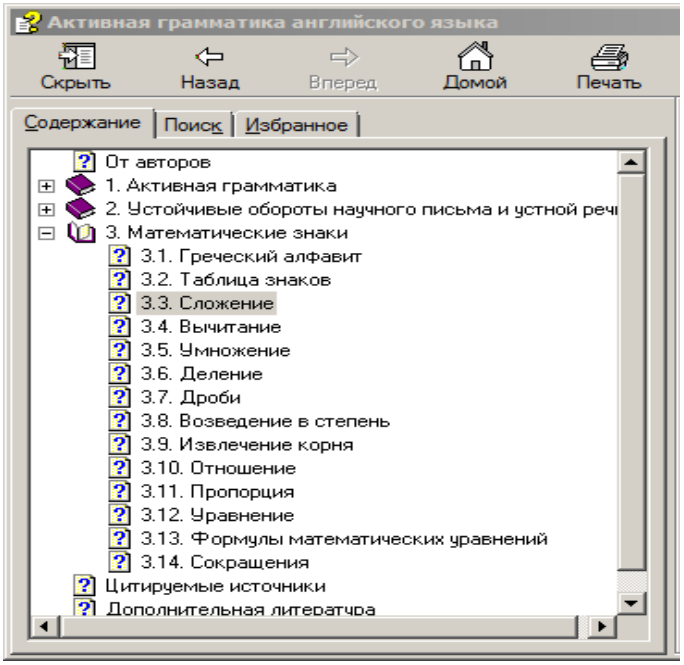

Рисунок 8. Оглавление учебного пособия

Применение формата CHM позволяет также упростить доступ к источникам, на которые ссылается автор (рис. 9).

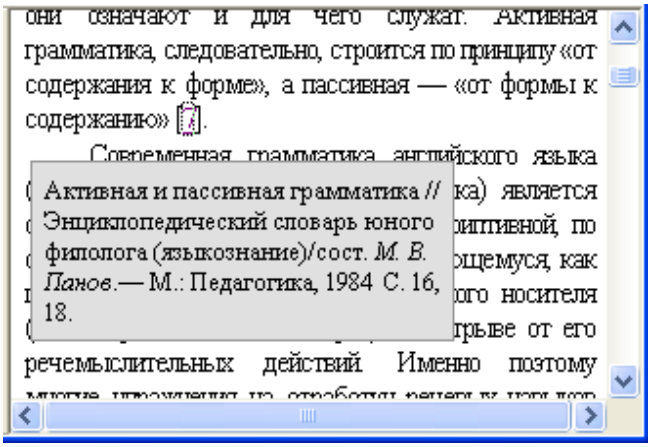

Рисунок 9. Интерактивные ссылки на источники в электронном пособии

Полезность интерактивного поиска в электронном пособии иллюстрирует рисунок 10.

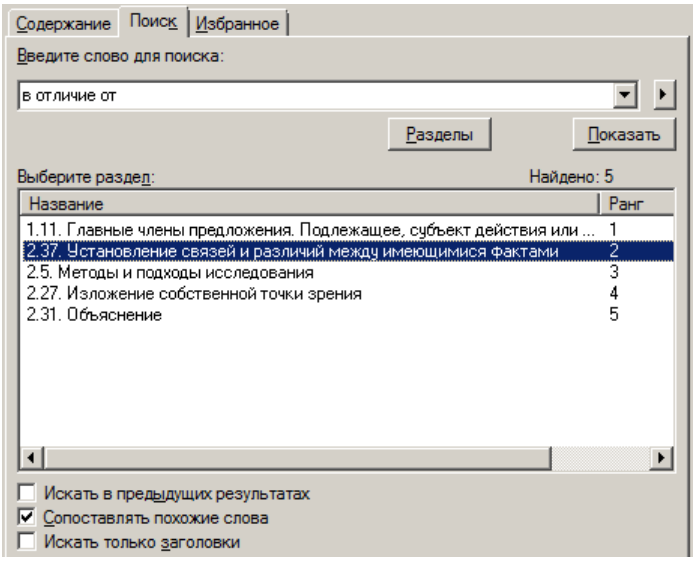

Рисунок 10. Поиск в электронном пособии

Разработанное пособие окажет существенную помощь студентам, аспирантам, изучающим английский язык, а также специалистам, подготавливающим статьи на английском языке и читающим лекции и технические доклады. Теоретическая база, идеологические основы которой заложены в работах [21, 23, 24], позволяет грамотно составлять сложные предложения, точно передавать смысл русских предложений, не теряя деталей (таких как завершённость действия, момент времени, отношение говорящего и т.д.), а богатый набор устойчивых конструкций позволяет сократить время, которое требуется для написания обязательных шаблонных фраз.

Первые попытки использовать ЭВМ для перевода текстов были предприняты в 1947 году в США, сразу после появления первых ЭВМ. Однако первая демонстрация возможностей машинного перевода состоялась только в 1954 году в США (Джорджтаунский эксперимент). Сразу после этого начались полномасштабные исследования в области машинного (автоматизированного) перевода в различных странах, прежде всего в Англии, Франции, ФРГ, Японии и в том же 1954 году в СССР. К середине 1960-х в США уже для практического использования были представлены сразу две системы русско-английского перевода: MARK (в Департаменте иностранной техники ВВС США) и GAT (разработка Джорджтаунского университета, использовалась в Национальной лаборатории атомной энергии и в центре Евратома, Италия).

В настоящее время существует множество коммерческих проектов машинного перевода. В России заметный вклад в развитие машинного перевода внесла группа под руководством проф. Р. Г. Пиотровского (ЛГПУ им. А. И. Герцена, Санкт-Петербург) [24,25, 26].

В то же время практически все специалисты и пользователи автоматизированных переводов отмечают их невысокое качество, в особенности при переводе длинных смысловых фраз. Вот почему продвижение новых идей в создании алгоритмов машинного перевода, являющихся одновременно естественной, составной частью всей информационно-коммуникационной инфраструктуры образования и научно-технической сферы, по-прежнему чрезвычайно актуальная задача [27, 28, 29].

Одно из возможных направлений исследований представлено в настоящей работе, которая обобщает опыт авторов как в преподавании собственно английского языка и чтения лекций на английском языке, так и в написании математических текстов и словарей и в создании электронных аналогов таких словарей [9].

Ограниченный объем статьи позволяет лишь в общих чертах описать принципы предлагаемого подхода без ущерба, впрочем, для их понимания и последующей реализации.

Грамматика всякого языка как инструмента общения, как правило, весьма строга. Авторы полагают, что формализация правил грамматики естественного языка с целью построения эффективных алгоритмов анализа и синтеза предложений на этом языке возможна и актуальна.

Цель данной части статьи заключается в том, чтобы предложить подход к алгоритмизации проблемы перевода предложений с английского языка в режиме восприятия фразы (анализа) и на английский язык в режиме синтеза.

Во время процесса перевода с русского на английский язык переводчик изначально не знает, что он будет произносить в следующее мгновение и в какой форме. Следовательно, его сознание всегда «держит наготове» весь речевой лексико-грамматический аппарат. То же самое имеет место и при восприятии той или иной фразы.

В предложении обязательно указывается субъект и называются его действие или признак, объект действия, обстоятельства, в которых происходит действие, а также каждый член предложения, если он выражен существительным, можно по смыслу обозначить определением.

Задача. *Формализовать последовательность действий, выполняемых человеком при анализе и синтезе предложений английского языка.*

Теоретические исследования. Для решения обозначенной выше проблемы и связанной с ней задачи авторами разработана специальная методика. Опишем порядок действий, изначально и коротко предложенный в работах [16, 30], более детально.

Вначале выделяются 10 групп грамматических задач, которые объединены в матрицу (таблица 4) и которые обязательно должен решать любой произносящий предложение. В первой колонке даны номера задач, во второй – сами задачи. Назовем ее *сводной матрицей речевых задач*.

Таблица 4

№	***Альтернативы***			
1	Простое	Главное	Придаточное[24]	
2	Повествовательное[24]	Вопрос	Вопрос к подлежащему[24]	Восклицательное
3	Схемы *(см ниже) 1 2 3 4 5*			
4	Безличное	Именное безличное	***There be***	
5	Модальный глагол[45]		Эквивалент модального глагола	
6	Прошедшее	Настоящее	Будущее[45]	Будущее в прошедшем[45]
7	Простое	Длительное[45]	Совершенное[45]	Длительное в совершенном[45]
8	Изъявительное	Побудительное	Сослагательное	
9	Действительный залог		Страдательный залог[45]	
10	Инфинитив	Причастие I	Причастие II	Герундий
Сокращенные обозначения задач в формулах предложений (см. примеры в данной работе)				
1	Прост	Главн	Прид[24]	
2	Пов[24]	?	?п[24]	!
3	Схемы *(см ниже) 1 2 3 4 5*			
4	Безл	ИмБезл	Сущ	
5	М[45]		ЭМ	
6	П	Н	Б[45]	Бп[45]
7	П	Д[45]	С[45]	ДС[45]
8	Из	Поб	Сосл	
9	Дз		Стр[45]	
10	Inf	P1	P2	G

Опишем строки этой таблицы:

1 – статус предложения: простое, главное, придаточное;

2 – тип высказывания:

повествовательное, вопросительное, вопрос к подлежащему, восклицательное;

3 – одна из пяти схем предложения (схемы даны и описаны в таблице 2);

цифры верхним индексом означают, что данные задачи могут быть реализованы только в указанных схемах;

4 – безличные конструкции (безличные, именные безличные и оборот **there be**);

5 – выражение модальности: модальный глагол, эквивалент модального глагола;

6 – время глагола: прошедшее, настоящее, будущее, будущее в прошедшем;

7 – действие глагола: простое, длящееся, совершенное, длящееся в совершенном;

8 – наклонение глагола: изъявительное, побудительное, сослагательное;

9 – залог: действительный или страдательный;

10 – выражение действия с помощью неличных форм глагола: инфинитива, причастия I, причастия II, герундия.

Вторая часть таблицы представляет собой набор сокращений, используемых в формулах предложений в примерах анализа и синтеза, приведённых в настоящей работе ниже. Для того чтобы читать схемы предложений, рассмотренных в качестве примеров ниже, следует находить ячейку верхней подтаблицы таблицы 1, соответствующую ячейке нижней

подтаблицы, в которой находится сокращение, встретившееся в формуле предложения.

В строке 3 упоминаются 5 схем, по которым русскоязычный обучающийся сможет построить английский перевод русского предложения, а также перевести с английского языка на русский (см. рис. 7). Дадим более подробные пояснения к схемам:

1. Предназначена для построения побудительных предложений (*делай, будь* и т.п.), причем при отрицании глагола-сказуемого (*не делай*) перед ним необходимо произносить ***do not*** (= ***don't***). Глагол всегда в форме инфинитива без ***to***.

2. Положительная (без отрицания глагола) схема; предназначена для повествовательных предложений, придаточных предложений, вопроса к подлежащему, только для действительного залога в настоящем простом и прошедшем простом временах. В настоящем простом времени глаголы принимают форму либо (а) **V** (инфинитива без ***to***), либо (б) **Vs** (в 3 - м лице, единственном числе). Глагол ***be*** употребляется в формах ***am*** (1 - е лицо, единственное число), ***is*** (3 - е лицо, единственное число), ***are*** (множественное число) настоящего и ***was*** (единственное число), ***were*** (множественное число) прошедшего времён.

3. Отрицательная схема с частицей ***not***; предназначена для повествовательных предложений, придаточных предложений, вопроса к подлежащему, но только для построения предложений с глаголом ***be***: ***am***(1- е лицо, единственное число), ***is*** (3 - е лицо, единственное число), ***are*** (множественное число); ***was*** (единственное число), ***were*** (множественное число).

4. Положительная и отрицательная схема; предназначена только для построения вопросительных предложений с глаголом ***be***: ***am*** (1 - е лицо, единственное число), ***is*** (3 - е лицо, единственное число), ***are*** (множественное число); ***was*** (единственное число), ***were*** (множественное число).

5. Положительная (без отрицания глагола) схема; предназначена для повествовательных предложений, придаточных предложений, вопроса к подлежащему. Для действительного и страдательного залогов во всех временах, причем в действительном залоге глагол нельзя употреблять в формах настоящего простого и прошедшего простого времен.

6. Отрицательная схема; предназначена для повествовательных предложений, придаточных предложений, вопроса к подлежащему. Для действительного и страдательного залогов во всех временах.

7. Положительная и отрицательная схема; предназначена только для построения вопросительных предложений в действительном и страдательном залогах во всех временах.

В 4 - й и 5 - й схемах глаголы употребляются в следующих формах во всех временах: **V** (простое действие), **Ving** (длительное действие и длительное в совершенном), $\mathbf{V_3}$**ed** (совершенное действие и в страдательном залоге во всех временах).

Поскольку примеры с применением активной грамматики уже были выше рассмотрены, рассмотрим примеры с применением пассивной грамматики.

С пассивной грамматикой эффективнее всего работать при непосредственном общении с преподавателем (носителем языка) или специально разработанной программой. Преподавателем записываются по очереди слова из предложения в последовательности, в какой они произносятся в реальной речи. При использовании компьютерных программ слова автоматически выводятся на экран. Рассмотрим конкретный пример процесса восприятия предложения «**Your task is reading the text**» и дадим его лексико-грамматический анализ.

Суть заключается в том, чтобы постепенно вычеркивать ненужные речевые задачи в таблице 4. Задача преподавателя заключается в том, чтобы по мере необходимости давать объяснения.

На доске записывается первое слово – **Your.**

Сначала идентифицируется слово – притяжательное местоимение со значением «**ваш**».

1. Первое слово **your**: в этом случае начинать снимать задачи следует со *схем предложений* (см. строку 3): 1, 3 и 5 нужно зачеркнуть.

Объяснение: 1 - я схема (побудительное предложение с глаголом в соответствующем наклонении) может начинаться только с глагола, 3 - я схема – с вопросительного слова, вопросительного словосочетания или глаголов am, is, are, was, were, have, has, had, 5 - я схема может начинаться с вопросительного слова, вопросительного словосочетания, вспомогательного или модального глагола.

Во второй строке зачеркиваются «**?**» (вопросительное предложение, схемы 3 и 5, см. объяснение в предыдущем предложении) и «***?n***» (вопрос к подлежащему, схемы 2 и 4, но начинаться оно может только с вопросительного слова или вопросительного словосочетания).

В строке № 1 зачеркивается «***Прид***^{24}» (со слова **your** нельзя начинать придаточное предложение – только с союза). В строке № 8 следует зачеркнуть «***Поб***^{1}» (побудительное наклонение).

Следует отметить, что ***нельзя*** зачеркнуть задачи в строке № 4: возможно употребление **независимого причастного оборота**, после которого возможны указанные предложения (безличное предложение, именное безличное предложение и оборот **there be**).

2. Добавляется второе слово **your + task.**

В данном случае ничего нельзя зачеркнуть.

3. Записывается третье слово **your task + is.**

Идентификация: ***is*** – глагол с функциями вспомогательного глагола, эквивалент модального глагола, смыслового глагола и глагола-связки.

На данном этапе можно зачеркнуть в строке № 4 «***Безл***», «***ИмБезл***» (безличное и именное безличное предложение с формальным подлежащим **it**) и «***Сущ***» (оборот **there be**, в котором **there** – формальное подлежащее).

Объяснение: перед глаголом **is** может быть произнесено только подлежащее (в данном случае не **it** и не **there**).

Модальность, выражаемая модальным глаголом («***М***[45]»), зачеркивается, поскольку **be** – не модальный глагол.

В строке № 6 обводится кружком настоящее время («***Н***»), а остальные – зачеркиваются.

В строке № 7 зачеркиваются «***совершенный вид действия***» и «***длительное в совершенном***», так как таким глаголом нельзя выразить данные виды действия.

В строке № 8 обводится изъявительное наклонение («***Из***») и зачеркивается сослагательное («***Сосл***»). Последнее зачеркивается потому, что в простом, главном предложении (см. строку № 1) в сослагательном наклонении не должно быть зачеркнуто время «***Бп***» (будущее в прошедшем – в строке № 6).

4. Записывается четвертое слово **your task is + reading**.

Идентификация и перевод: **Ving** в предложении может быть произнесено в качестве причастия1 или герундия. **Be** подтверждается как глагол-связка, а **reading** – как герундий в качестве ИЧС, поэтому в строке № 7 зачеркивается длительный вид действия («***Д***»).

В строке № 8 необходимо зачеркнуть страдательный залог и обвести действительный. В пятой колонке необходимо зачеркнуть эквивалент модального глагола («***ЭМ***»), поскольку не был произнесен инфинитив после **is**. Зачёркивается 4-я схема, поскольку глагол **be** является связкой, а в 4-й схеме в форме **is** он может быть произнесен только в функции вспомогательного глагола.

5. Записывается пятое слово **your task is reading + the text.**

Поскольку предложение закончилось, можно зачеркнуть «***главное предложение*** (***Главн***)» в строке № 1, восклицательное предложение («!») в строке № 2, а в строке № 10 обвести герундий, а остальные зачеркнуть.

Полная формула предложения по окончании разбора выглядит так:

Прост Пов 2 НП Из ДЗ // ИЧС-G

Расшифровка: простое предложение, повествовательное, 2-я схема; глагол в форме настоящего простого времени, в изъявительном наклонении, залог – действительный; именная часть сказуемого выражена герундием.

Алгоритмизация

При решении задачи алгоритмизации предложенного метода (для автоматического или автоматизированного анализа предложений) ключевыми являются следующие требования:

- метод должен включать формализованное и структурированное описание множества вариантов;
- метод должен включать набор формальных или формализуемых правил исключения подмножеств множества вариантов по мере рассмотрения последующих слов предложения;
- для автоматического анализа нужно принимать в расчёт тот факт, что результатом определения формы слова и грамматической информации,

соответствующей ей, чаще всего является не один, а несколько возможных вариантов разбора (алгоритм должен по мере возможности устранять морфологическую омонимию: например, слово ***wish*** английского языка анализатор форм слов определит как «либо глагол, либо существительное», а анализатор синтаксиса в предложении ***I wish you to hear my wish*** устранит омонимию, опираясь на контекст).

Перед тем как перейти к рассмотрению возможных подходов к формализации и алгоритмизации метода, следует отдельно рассмотреть место синтаксического анализа в процессе анализа текста на естественном языке. Системы для работы с естественным языком, как правило, включают в себя несколько модулей, последовательно осуществляющих анализ языка на различных уровнях структурной организации.

Как в отечественной [31], так и в зарубежной [32] литературе, в задаче анализа текста на естественном языке различают (в порядке решения задач) морфологический, синтаксический, семантический и прагматический уровни анализа текстов на естественных языках. В задаче синтеза порядок следования модулей заменяется обратным (рис.11).

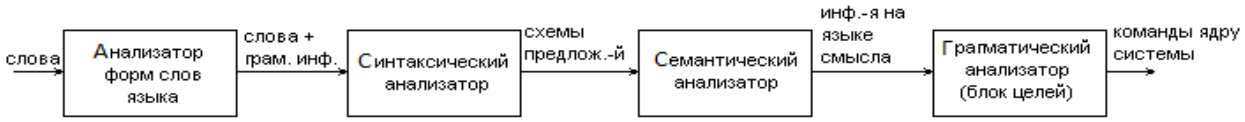

Рисунок 11. Уровни анализа текста на естественном языке

Формально рассматриваемый в данной статье метод в качестве входных данных принимает не текст на естественном языке, а последовательность лексем, получаемую в результате анализа форм слов рассматриваемого предложения. Поэтому, если ставить целью разработку полностью автоматического анализатора предложений, следует сначала выбрать или создать модуль, выполняющий анализ форм слов языка (см., например, [33]), а затем опираться на формат выходных данных этого анализатора в качестве формата входных данных.

Для построения автоматизированной системы анализа предложений следует также рассмотреть вариант с интерактивным процессом расстановки грамматических тегов (пометок с грамматической информацией) над словами рассматриваемого предложения непосредственно перед анализом предложения или заранее.

Помимо этого, для построения алгоритма анализа предложения необходимо, чтобы информация о множестве вариантов трактовки предложения была представлена в максимально структурированной форме, допускающей эффективное выделение подмножеств с целью их снятия (исключения из рассмотрения).

Множество возможных вариантов трактовки предложения можно структурировать различными способами, например:

– в виде множества листьев дерева вариантов анализа предложения;

– в виде многомерного пространства, векторами которого являются конкретные варианты трактовки предложений, а измерениями – грамматические категории языка.

Для реализации алгоритма анализа предложения на естественном языке необходимо построение системы грамматических категорий, характеризующих как можно более полно максимально возможное число (по возможности все) предложений языка.

Кроме того, для работы алгоритма необходима информация о правилах снятия вариантов трактовок предложения. В рамках данной работы предполагается, что правила должны иметь вид «*если начало предложения соответствует некоторому шаблону X, то следует исключить из рассмотрения трактовки предложения из множества M**».

Для формализации набора правил следует также

– принять во внимание особенности формата входных данных (слов с соответствующей им грамматической информацией, возможно в нескольких вариантах): в ходе анализа потребуются уточнения грамматической информации, сопоставленной анализатором форм слов отдельным словам, в частности на этом уровне возможно частичное устранение омонимии;

– разработать структуру данных, обеспечивающую хранение правил;

– разработать структуру данных, служащую для представления текущего состояния системы.

Опишем модель формально:

– пусть есть множество M всех возможных трактовок всех предложений языка. Это множество, очевидно, очень велико; очевидно также, что оно конечно. Вопрос о наиболее оптимальной его структуризации мы оставляем открытым, однако придерживаемся описанной ранее многомерной модели [измерениями будем считать грамматические категории (например, *наклонение* или *время*), а компонентами векторов – значения (наклонение – *побудительное,* время – *прошедшее*) категорий]. Окончательная трактовка предложения будет представлять собой точку пространства (или малое его подмножество, если предложение может быть трактовано несколькими способами);

– *грамматический маркер* W, описывающий слово, – это результат морфологического анализа этого слова, содержащий возможные варианты его морфологического разбора (их может быть несколько) – код основы слова, морфологические признаки (число, падеж…); в общем случае $W= =(w_1,.., w_n)$ – набор всех морфологических омонимов слова;

– *входные данные* для анализа – это цепочка грамматических маркеров $S= = (W_1..W_n)$, составляющих предложение;

– неполный маркер ω есть такой грамматический маркер, который описывает не одно конкретное слово языка, а некоторый класс слов. Это достигается путём указания неполной карты морфологических признаков или указания класса допустимых основ слов вместо конкретной основы слова;

– *правило* r (Σ, M*) снятия вариантов – это пара множеств неполных маркеров $\Sigma=(\sigma_1,..,\sigma_n)$ и вариантов трактовок M*, однозначно исключаемых из

рассмотрения, если начало предложения соответствует набору Σ. Множество всех правил R описывается грамматикой языка. *Порядком правила* | r.S | будем называть число неполных маркеров, входящих в Σ.

Применение правила приводит к сокращению множества допустимых вариантов трактовок предложения (M’ = M \ M*).

Правила однозначны; применение означает полное и окончательное исключение из рассмотрения вариантов трактовки, перечисленных в теле правила, что затрудняет составление свода правил для естественного языка.

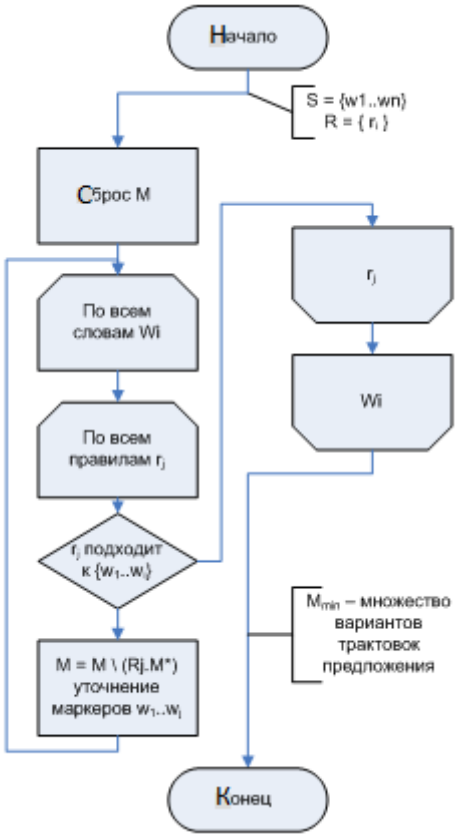

Рисунок 12. Блок-схема алгоритма анализа

Укрупнённая схема алгоритма анализа (рисунок 12) демонстрирует порядок выполнения предлагаемого алгоритма, реализующего метод анализа предложений на естественном языке.

Обозначения, применённые на схеме:

r_j.M* – множество вариантов трактовок предложения, исключаемое в соответствии с правилом r_j.

Под уточнением маркеров $w_1..w_n$ понимается снятие вариантов маркеров, не соответствующих правилу (только в случае морфологических омонимов).

M = M – r_j.M* – из множества M всякий раз удаляются элементы, содержащиеся во множестве r_j.M*.

Множество R всех правил считаем упорядоченным по возрастанию *порядка правил*, входящих в него.

Приведённая схема алгоритма является *упрощённой*, поскольку реальный анализ претерпевает ветвление всякий раз, когда возможно применение исключающих друг друга правил вследствие морфологической омонимии в словах предложения. В этом случае параллельно рассматриваются оба варианта усечения множества М.

Для данного алгоритма, вместе с тем, решения ряда проблем остаются открытыми, затрудняя тем самым непосредственную реализацию его на языке программирования. В первую очередь это связано с большим объёмом работ по внесению значительного количества правил, описывающих синтаксис английского языка:

– на данный момент выделение грамматических категорий, отвечающих заданным в статье критериям (существование и единственность значения категории для каждой из категорий, достаточность набора категорий для описания любого или подавляющего большинства предложений языка), в полной мере ещё не выполнено, но, тем не менее, авторы главы не видят принципиальных препятствий к выбору таких категорий или синтезу суррогатных категорий, представляющих собой объединения обычных грамматических категорий, используемых в лингвистике;

– в работе не приводятся формальные правила исключения вариантов ввиду того, что полный набор формальных правил будет иметь большой объём, не соответствующий объёму публикации.

Помимо этого, описанный алгоритм не раскрывает детали обработки данных, а описывает саму идею выполнения анализа, поэтому авторы главы продолжают работу по улучшению и модернизации предложенного алгоритма по перспективным направлениям:

– оптимизация на уровне представления множества вариантов: возможен переход к первой из предложенных моделей структуризации множества возможных вариантов (древовидной) в том случае, если дерево вариантов трактовки предложения окажется возможным построить автоматически, опираясь, к примеру, на предложенную в статье многомерную модель (или, возможно, на какую-то другую модель), – переход к древовидному представлению позволит, во-первых, расширить базис операций для формирования условий снятия вариантов в правилах, а во-вторых, сократить количество правил, требующих проверки на каждом шаге анализа;

– оптимизация на уровне представления правил: с одной стороны, усложнив структуру правил, можно добиться меньшего числа проверок [например, настроив (или определив) бинарное отношение «А строже Б, если тождественно истинен предикат (для любого x) $A(x) \rightarrow B(x)$ (где $\rightarrow$ есть символ импликации, обратная импликация не столь важна)» между правилами или кэшируя результаты проверки применимости правил вида «предложение начинается с **Do you**..», результат которой остаётся неизменным после анализа второго слова предложения], с другой, упростив структуру правил, можно добиться сокращения времени проверки применимости каждого из них.

Помимо значимости описанного подхода как подхода к решению задачи автоматизации анализа текста на естественном языке, следует также отметить, что предложенный метод реализует новый подход в объяснении грамматического и лексического материала. Многие известные на данный момент методики обучения иностранному языку подразумевают прохождение, объяснение и изучение лексико-грамматического материала от частного к общему, а предложенный алгоритм – от общего к частному.

Идеи, касающиеся метода обучения людей анализу предложений, изложенные в данной главе, ранее были предложены научному сообществу на международной конференции [30].

Ниже даны некоторые практические шаги для разработки компьютерной программы по обучению студентов владению пассивной грамматикой на основе сводной таблицы речевых задач.

На рисунках 13-24 показан возможный внешний вид оглавления электронного пособия по обучению пассивной грамматике. Также на примере конкретного предложения будет показан принцип работы обучающей программы. Рассмотрим разбор английского предложения «When were they invited to this party?» в режиме пословного предъявления на экран, причем разберем его с возможной ошибкой. Задача пользователя - поставить в пустых квадратиках значок ***v***, что означает снятие той или иной грамматической задачи с появлением каждого последующего слова. После того как все значки ***v*** поставлены пользователем (с его точки зрения), нужно нажать кнопку «Выполнить». Если значки снятия задач поставлены в квадратах правильно, то должно появиться следующее слово.

Шаг 1. В окне программы со списком задач появляется слово **When**:

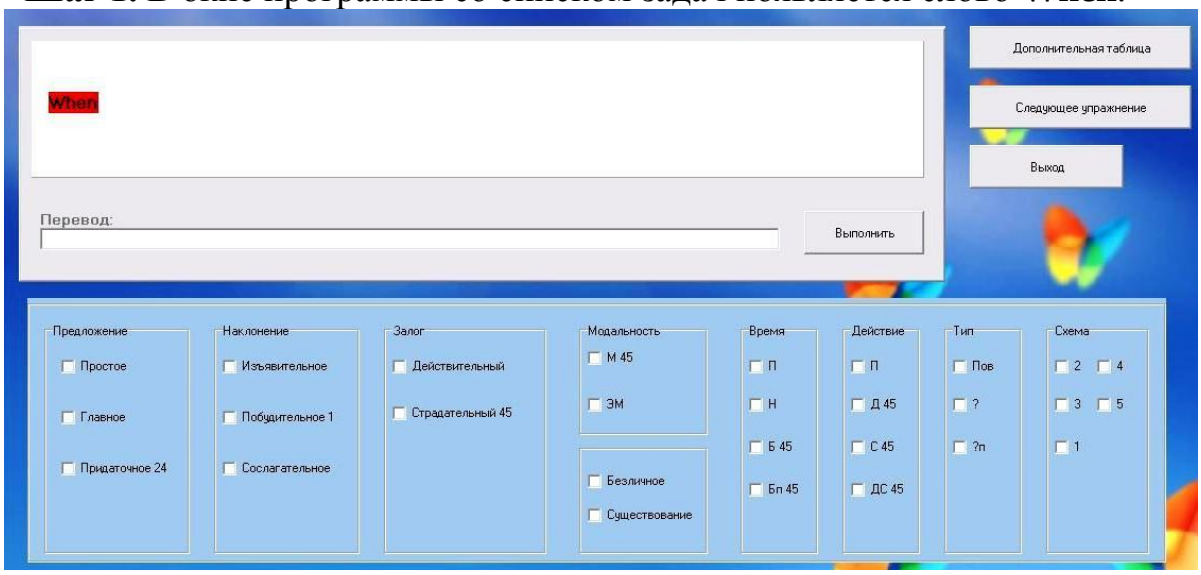

Рисунок 13. Шаг 1

Шаг 2. Предположим, что учащийся снял следующие задачи (см. рисунок 14):

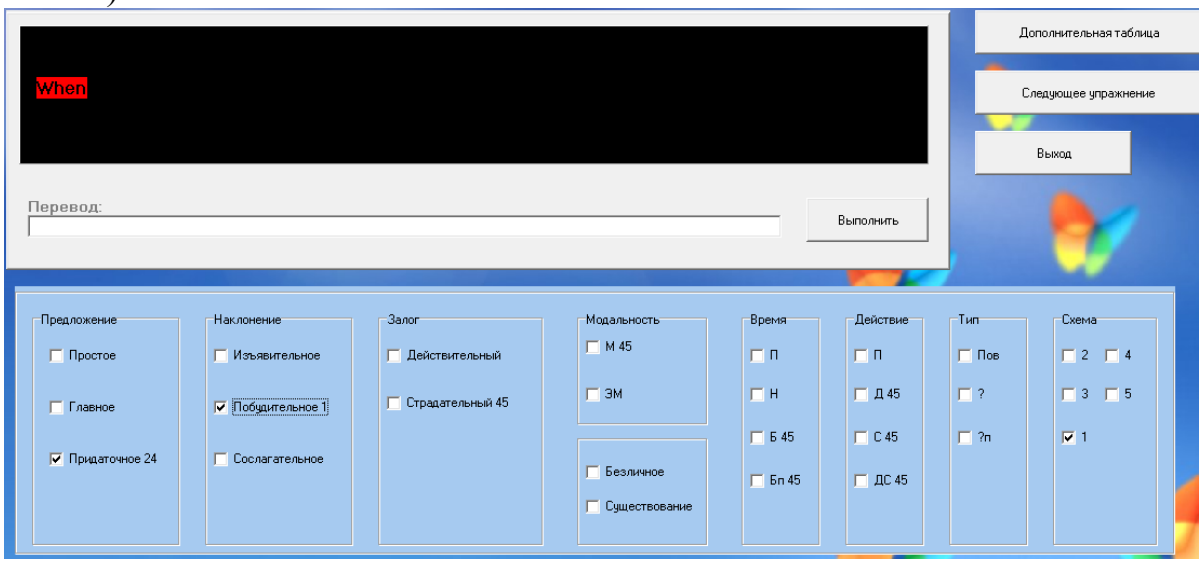

Рисунок 14. Шаг 2

Шаг 3. После нажатия клавиши «Выполнить» появляется окно с текстом «Вы дали неверный ответ. Начните сначала». Это означает,что задачи или одна из задач сняты неверно и необходимо сделать еще одну попытку или попытки. Естественно, необходимо разобрать ошибки. В данном случае задачу «Придаточное предложение» снимать еще рано, поскольку «when» может начинать придаточное предложение времени или придаточное подлежащее. Кроме того, задачу «вопрос к подлежащему» (?п) необходимо снять.

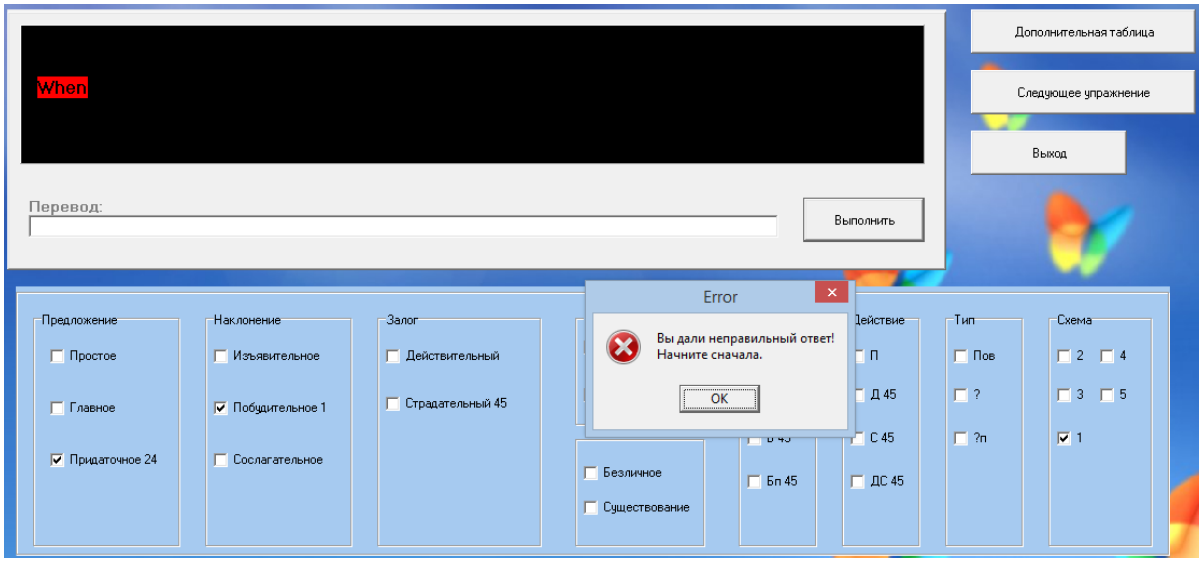

Рисунок 15. Шаг 3

Шаг 4. Повторяется операция с учетом допущенных в предыдущем шаге ошибок.

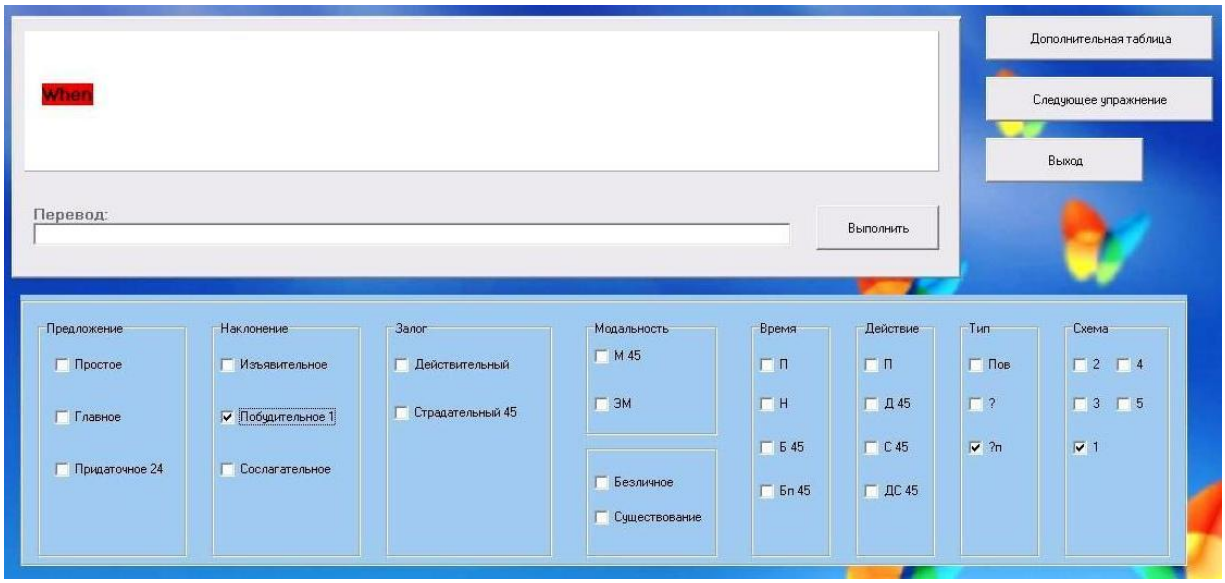

Рисунок 16. Шаг 4

Шаг 5. Нажимаем «Выполнить». Появляется слово **were** с указанием на все его функции в предложении (рис. 17). Значит, предыдущий шаг выполнен верно.

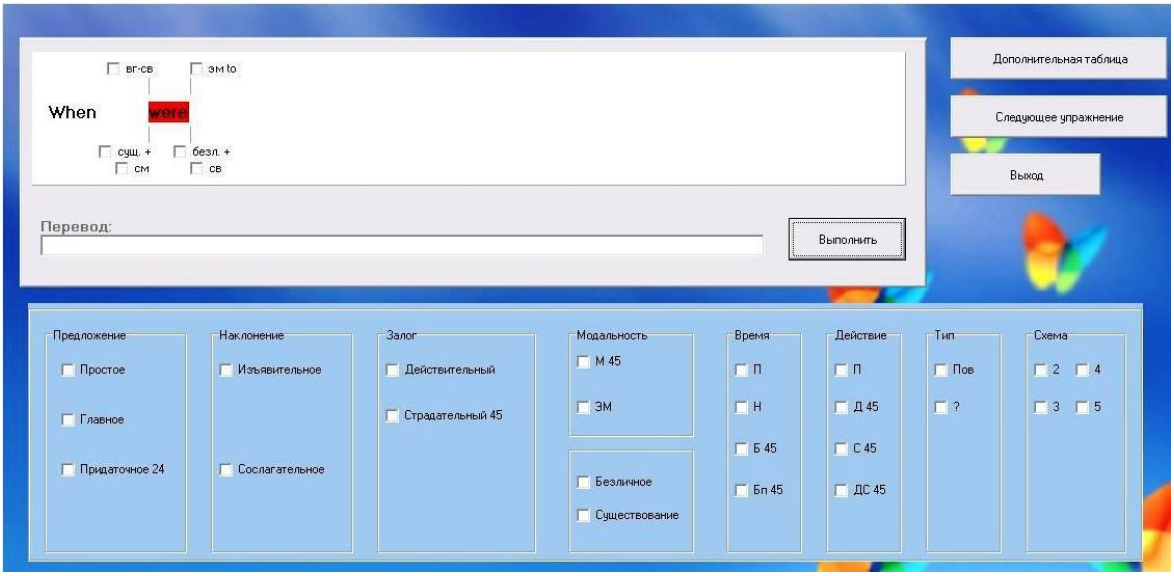

Рисунок 17. Шаг 5

Шаг 6. Снимаем задачи. **Were** есть глагол, не модальный (снимается $М^{45}$), указывает на прошедшее время (соответственно снимаются Н, Б, Бп), на два вида действия – простой и длительный (снимаются С и ДС), на изъявительное наклонение (снимается сослагательное). Задача «Безличное предложение» снимается, поскольку форма **were** указывает только на множественное число, а в безличном предложении формальное подлежащее

it указывает на единственное число. Событие не стало развиваться по схеме 2 и 4 (снимаются), а значит, предложение не придаточное и не повествовательное.

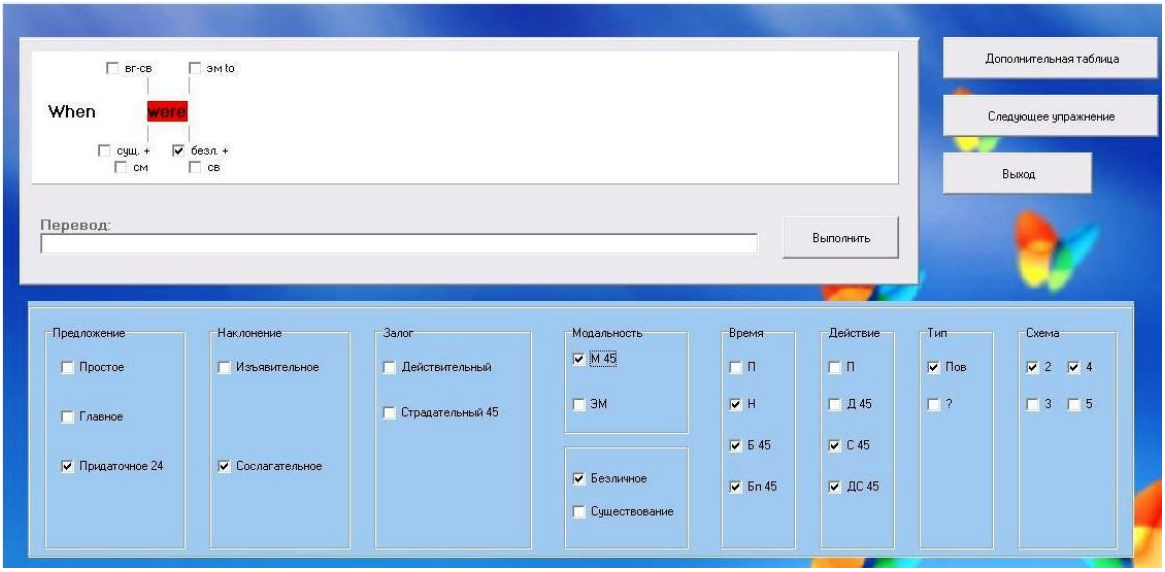

Рисунок 18. Шаг 6

Шаг 7. Нажимаем «Выполнить». Открывается слово **they,** и переходим к следующему шагу.

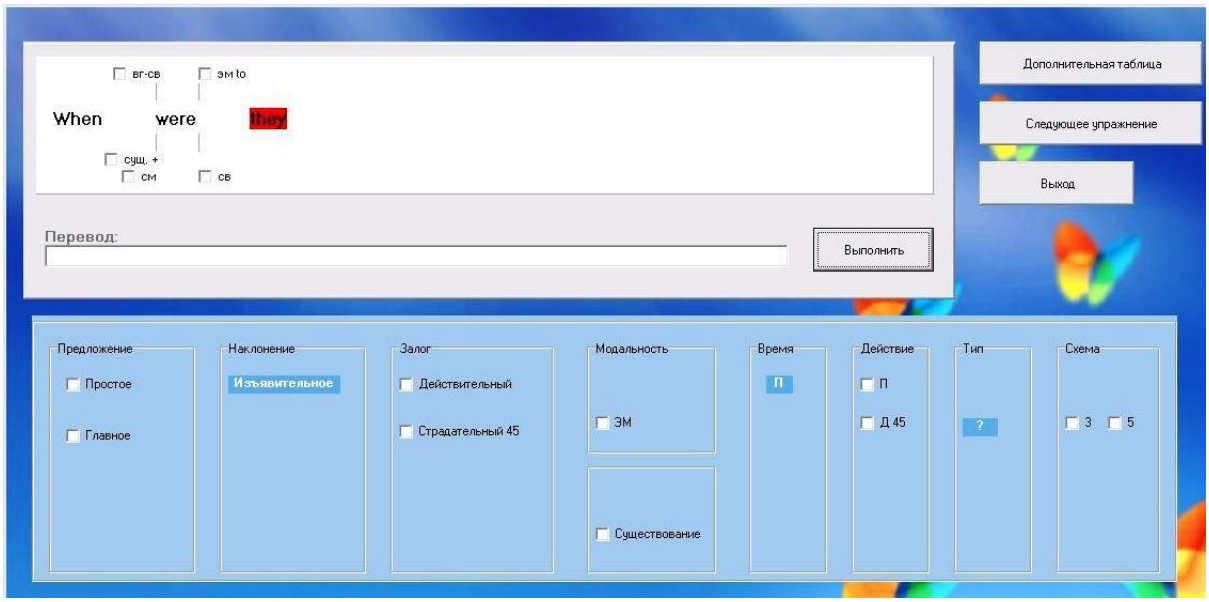

Рисунок 19. Шаг 7

Шаг 8. Мы можем снять только одну задачу – существование чего-либо (оборот **there be**), поскольку после глагола должно было появиться **there**.

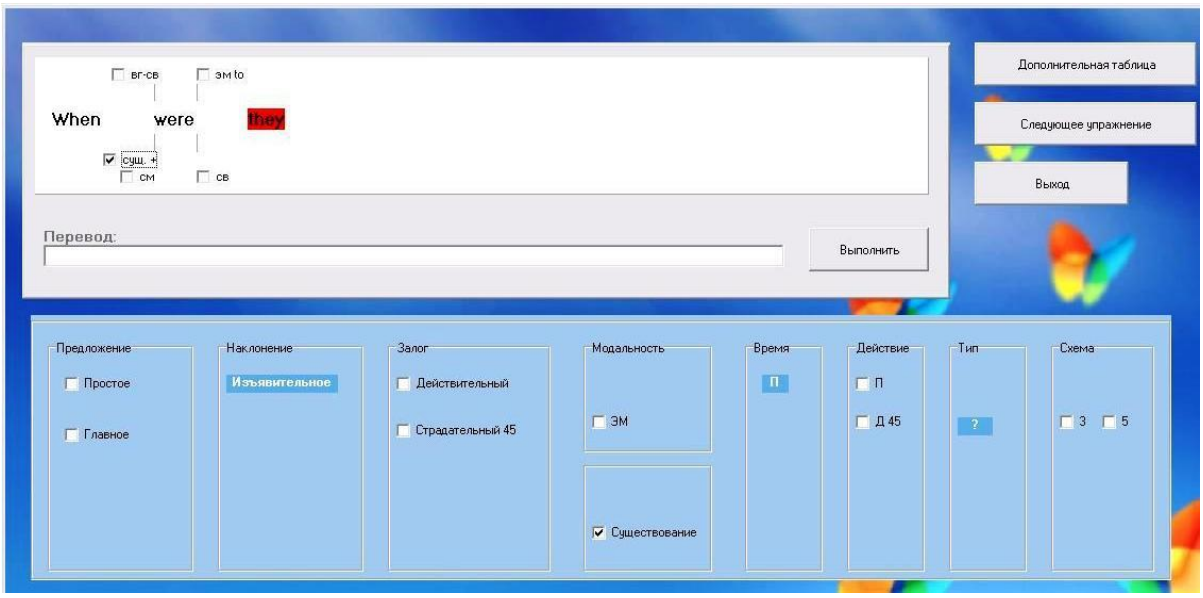

Рисунок 20. *Шаг* 8

Шаг 9. Нажимаем «Выполнить». Появляется слово **invited**. Значит задачу сняли правильно.

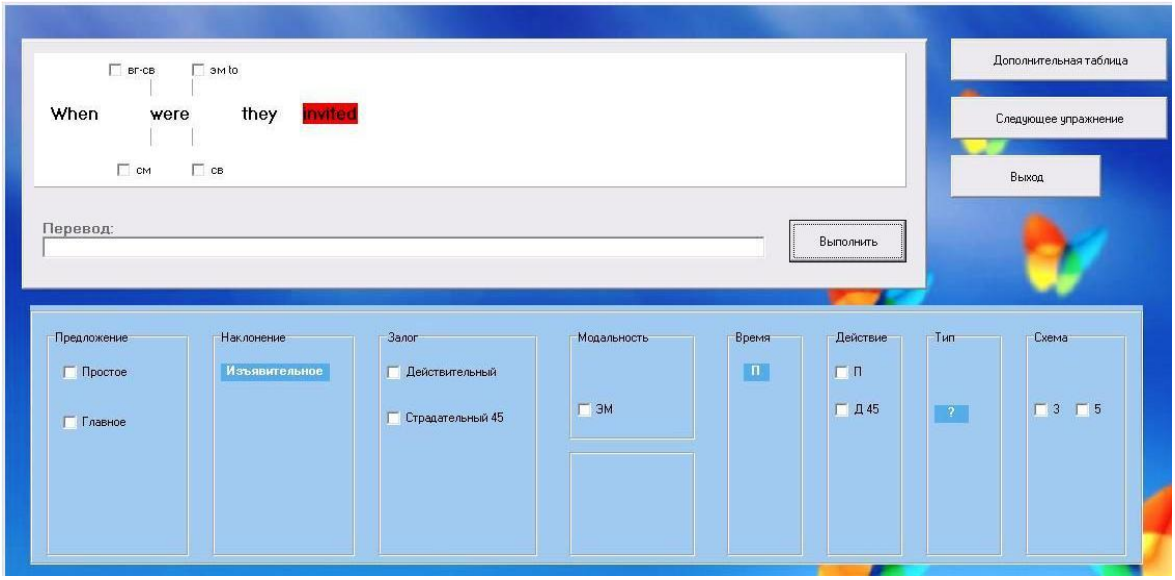

Рисунок 21. *Шаг* 9

Шаг 10. По форме смыслового глагола **invited** определяем, что глагол were является вспомогательным (остальные функции снимаем), а смысловой глагол употреблен в форме страдательного залога. Вид действия простой, схема 5.

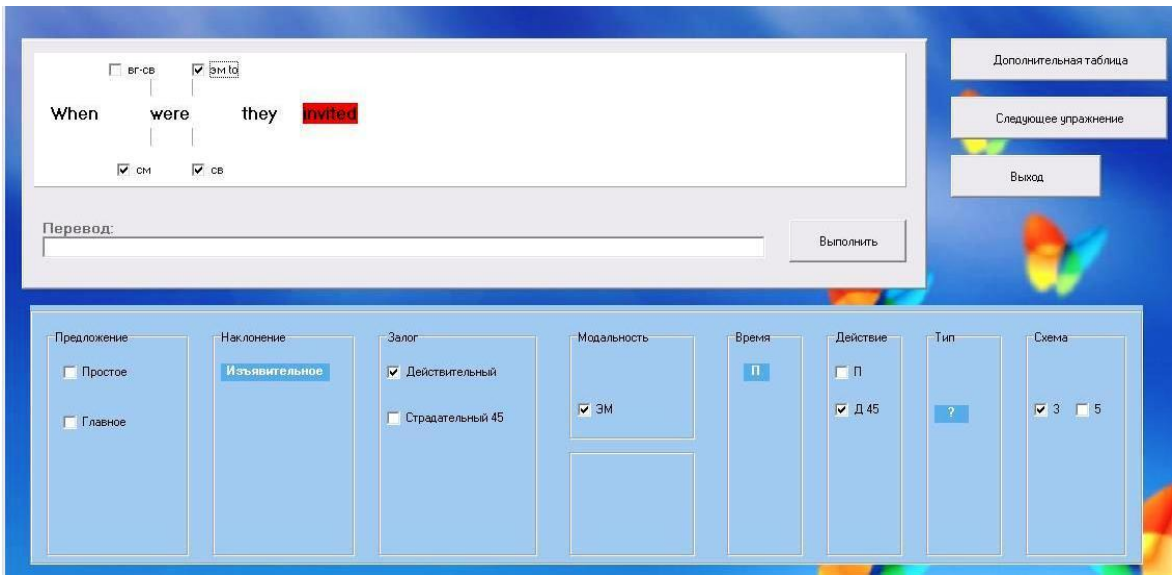

Рисунок 22. *Шаг* 10

Шаг 11. Нажимаем «Выполнить». Появляются слова **to this party?**.

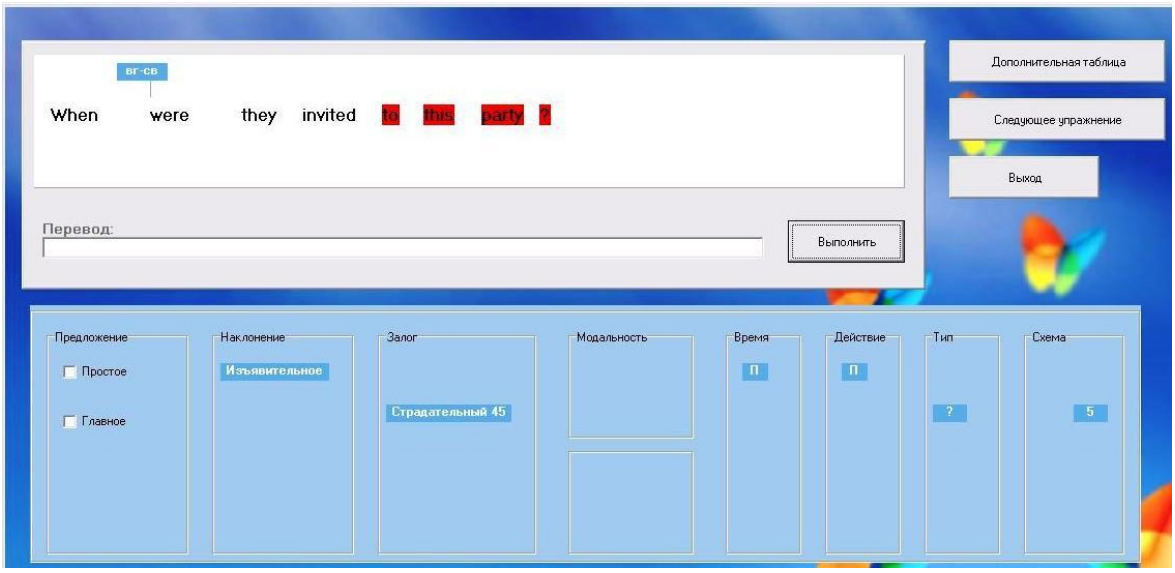

Рисунок 23. *Шаг* 11

Шаг 12. Снимаем задачу «главное предложение» и нажимаем «Выполнить». В строку «Перевод» по необходимости можно набивать текст перевода.

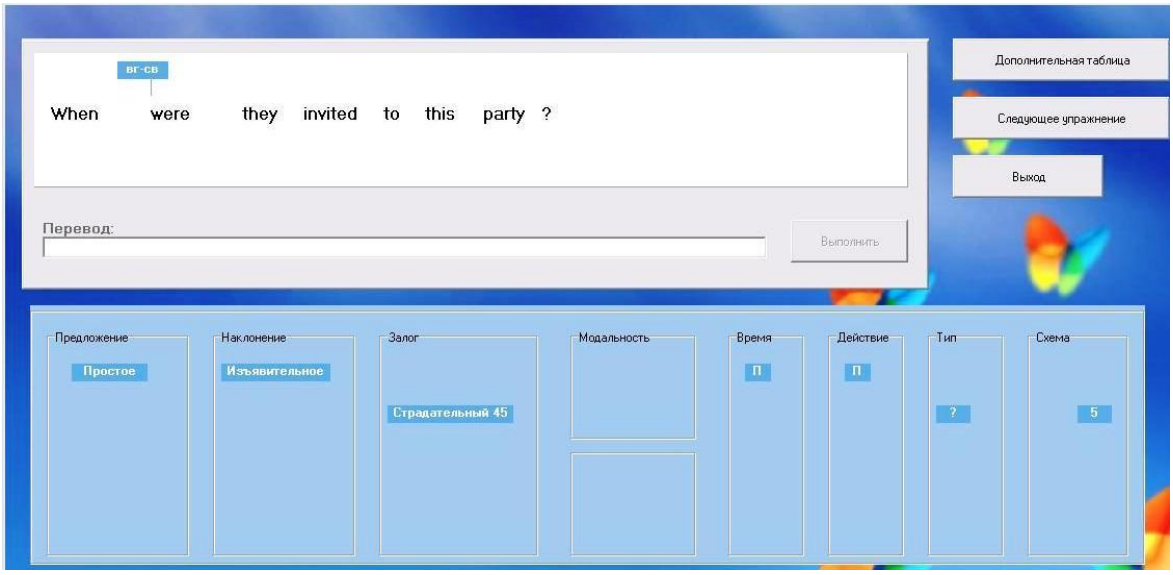

Рисунок 24. *Шаг* 12

Заключение

Таким образом, в развитие исследований авторов [34], в статье раскрыты вопросы происхождения языка, выведена всеобщая формykf познания и языка, выведены основные грамматические категории языка, а также выведена грамматическая формула предложения. Дано теоретическое обоснование активной и пассивной грамматик и дано описание практического применения в обучении студентов «живым» способом и с помощью прикладных компьютерных программ.

Литература к главе 2

1. Философский энциклопедический словарь / Гл. редакция: Л.Ф. Ильичев, П.Н. Федосеев, С.М. Ковалев, В.Г. Панов. М.: Сов. энциклопедия, 1983. 840 с.
2. Современный философский словарь. СПб.: Академический проект, 2004. 864 с. ISBN 5-8291-0364-8
3. Энгельс Ф. Диалектика природы. Заметки и фрагменты. К. Маркс и Ф. Энгельс Сочинения, изд. 2, Т. 20
4. Chimps use touches and noisy gestures when trying to get another chimps attention, researcher finds. http://phys.org/news/2013-01-chimps-noisy-gestures-attention.html
5. Когда ребенок говорит первое слово. http://www.privivki-detyam.ru /pervie-slova-rebenka.-kogda-rebenok-govorit-pervoe-slovo.html
6. Шанский Н.М., Боброва Т.А. Школьный этимологический словарь русского языка. М.: Дрофа, 2002.
7. Объективная диалектика. Т. 1. отв ред. тома Ф.Ф. Вяккерев. М.: «Мысль». 1981. Материалистическая диалектика в пяти томах;. под общей редакцией Ф.П. Константинова, В.Г. Марахова.
8. Энгельс Ф. Роль труда в процессе превращения обезьяны в человека. К. Маркс и Ф. Энгельс. Сочинения, изд. 2, Т. 20, с. 486-498.

9. Заволокин А.И. Всеобщая формула познания и ее использование при изучении иностранного языка. Лингвострановедение: методы анализа, технологии обучения Тринадцатый межвузовский семинар по лингвострановедению. (Москва, 16-17 июня 2015 г.) сборник научных статей в 2 частях. Часть I. Языки в аспекте лингвострановедения / Московский государственный институт международных отношений (университет) МИД России; (под общей редакцией Л.Г. Ведениной). М: МГИМО–Университет, 2016. 371 с.
10. http://yunc.org/активная_и_пассивная_грамматика
11. Заволокин А.И., Миронов В.В. Активная грамматика английского языка (для физико-математических и инженерно-технических специальностей). М.: Горячая линия – Телеком, 2015.
12. Заволокин А.И., Миронов В.В. Активная грамматика английского языка (к языковой автоматизации): учеб. пособие для вузов. 2-е изд.М.: Горячая линия – Телеком, 2015. 240 с: ил.
13. Заволокин А.И. Лингвопсихофизиологический аспект обучения студентов владению речью в режиме перевода с иностранного языка на родной в сб.: Язык и коммуникация в контексте культуры: материалы международной научно-практической конференции, 21-22 марта 2008= Third Annual International Conference. Ryazan State University, Russia/ отв. ред. С.В. Лобанов; Ряз. гос. ун-т им. С.А. Есенина. Рязань, 2008. 236 с. С.176-182.
14. Заволокин А.И. Обучение синхронному переводу студентов языковых и неязыковых вузов при помощи "сводной таблицы речевых задач" в сб.: Проблемы преподавания профессионально-ориентированного иностранного языка: материалы междун. науч.-практ. конф. / [сост.Е.Е. Сухова]. Рязань: Издатель Ситников, 2008. С. 159-169.
15. Заволокин А. И. Философский и психолингвистический взгляд на проблемы языка и речи и на проблемы создания активной и пассивной грамматики Наука и образование XXI века: материалы IV Международной научно-практической конференции (29 октября 2010 г., СТИ, г. Рязань). В 4-х томах: Том 4. Часть 1. Современные проблемы образования; / под общей редакцией проф. А.Г. Ширяева, доц. А.В. Барановского. Рязань, СТИ, 2010.98 с.
16. Заволокин А.И., Миронов В.В. Алгоритмическая процедура перевода с использованием активной грамматики //Сб. статей междун. науч.-практ. конф. «Актуальные проблемы современных лингвистических исследований». Рязань: Изд. «Узорочье», 2012. С. 277-283.
17. Леонтьев А.А. Основы психолингвистики. М., 1997.287 с.
18. И.П. Сусов. История языкознания: учеб. пособие. Тверь: ТвГУ, 1999.
19. Активная и пассивная грамматика // Энциклопедический словарь юного филолога (языкознание) /сост. М.В. Панов. М.: Педагогика, 1984 . с.: 16,18 // http://www.pedlib.ru/Books/1/0088/1_0088-1. shtml
20. Бабайцева В.В., Чеснокова Л.Д. Русский язык: теория: учеб. для 5-9 кл. общеобразоват. учреждений. 4-е изд., дораб. М.: Просвещение, 1995. 256 с.
21. Миронов В.В, Заволокин А.И., Розанов А.К. Проблема формализации правил русско-английского и англо-русского переводов текстов Научно-методический журнал «Информатизация образования и науки» № 2(22) / 2014.

http://informika.ru/pechatnye-izdaniya/zhurnal-informatizaciya- obrazovaniya-i-nauki/arhiv-vypuskov/2014/vypusk-n22-soderzhanie/

22. Пруцков А.В. Алгебраическое представление модели формообразования естественных языков / Электронный журнал Cloud of Science. 2014. Т.1. №1.

23. Пиотровский Р. Г. Компьютеризация преподавания языков. Л.:ЛГПИ, 1988.

24. Пиотровский Р.Г. Автоматическая переработка текста: теория и практика к концу XX в. Л:. ЛГПУ,. 2002.

25. Пиотровский Р. Г. Системно-семиотический компьютерный анализ и синтез единиц языка и речи. 07-04-90402а / Б. РосГПУ. 2008.

26. Бухенский К.В., Заволокин А.И., Миронов В.В., Розанов А.К. Информационная система «Русско-английский словарь математических терминов»/ Изд. 1.2 (исправленное и дополненное), М.: РАО, Объединенный фонд электронных ресурсов «Наука и образование». 2013. Рег. № 18951. Объем - 7,2 Мб.

27. Кулагина О.С. О современном состоянии машинного перевода // Математические вопросы кибернетики. 1991. Вып. 3. С. 5-50.

28. Куракин Д.В. О разработке моделей и системных проектных решений для информационно-коммуникационной инфраструктуры управления научно-технической сферой //Информатизация образования и науки. 2012. № 2 (14). С. 68-78.

29. Заволокин А.И. Проблема разработки активной и пассивной грамматики для обучения владению иноязычной речью. Язык и коммуникация в контексте культуры/ Материалы IV междун. науч.-практ. конф. Рязань: ООО «ПРИЗ-Р», 2009. С. 193-199.

30. Белоногов Г.Г., Богатырев В.И. Автоматизированные информационные системы /под ред. К.В. Тараканова. М.: Сов. радио, 1973 328 с.

31. Jurafsky D., Martin J.H. Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. //Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 2000, 950 p.

32. Пруцков А.В. Генерация и определения форм слов естественных языков на основе их последовательных преобразований // Вестник Рязанского государственного радиотехнического университета. Рязань, 2009. Вып. 27. С. 51-58.

33. Valentin V. Mironov, Alexandr I. Zavolokin and Aleksey K. Rozanov. Preparing electronic handbook for using active grammar during process of translation of technical texts into English. SHS Web of Conferences **9**, 02029 (2016). http://www.shs-conferences.org /articles/shsconf /abs/2016/07/contents /contents.html

Глава 3

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ КОНТРОЛЯ И ПОИСКА ДАННЫХ

Часть 1. СИСТЕМА КОНТРОЛЯ ЗНАНИЙ ПО МОРФОЛОГИИ

1. Назначение системы

Система контроля знаний по морфологии Salvinia Examiner является вспомогательным средством для обучения морфологии русского языка. Этот комплекс обучающего и контролирующего программного обеспечения позволяет решать следующие задачи в обучении:

- формирование умения определять грамматическую информацию для написанных слов (распознавать число и падеж имён существительных, время глагола и т.д.);
- формирование навыка построения требуемых форм слов (например, умения просклонять некое существительное).

2. Компоненты системы

Система контроля знаний Salvinia Examiner включает в себя следующие компоненты:

- Salvinia Editor – редактор баз знаний о морфологии языка. В этом приложении инженер по знаниям (преподаватель) может расширять базу знаний об изучаемом языке (рис. 1,2);
- Salvinia Test Editor – редактор заданий для проведения тестирования (рисунок 3) – позволяет составлять файлы заданий, в рамках которых приложение-тестер проводит сеансы тестирования людей, изучающих морфологию данного языка;

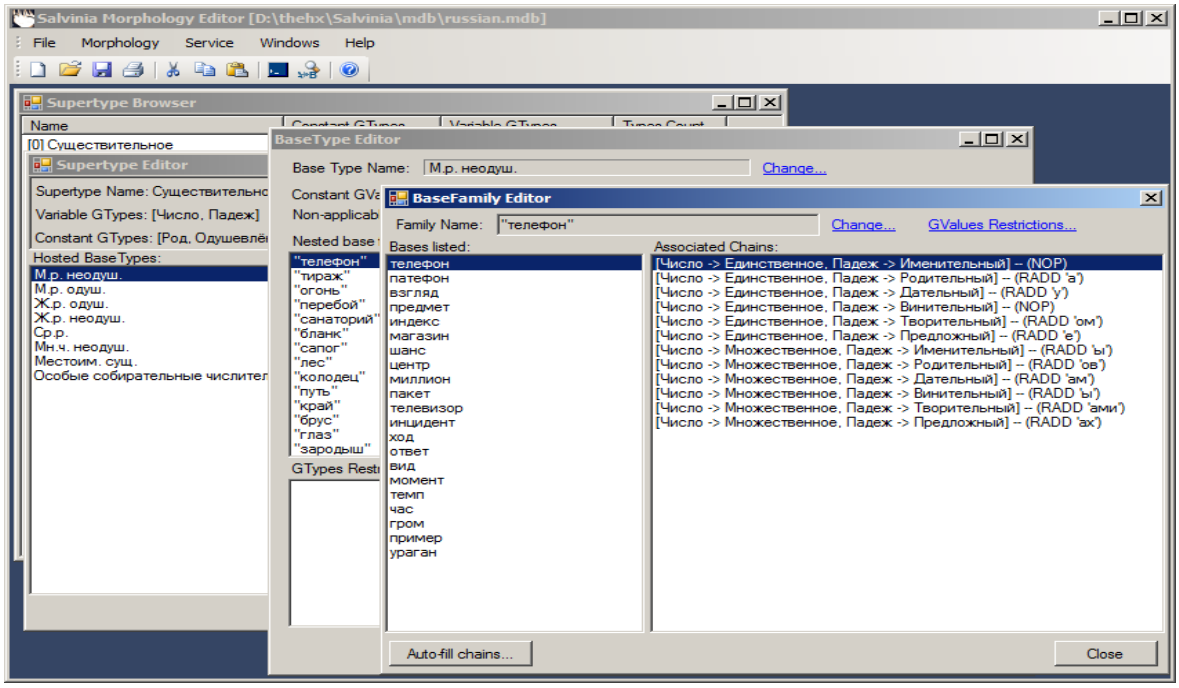

Рисунок 1. Редактирование базы знаний о морфологии языка

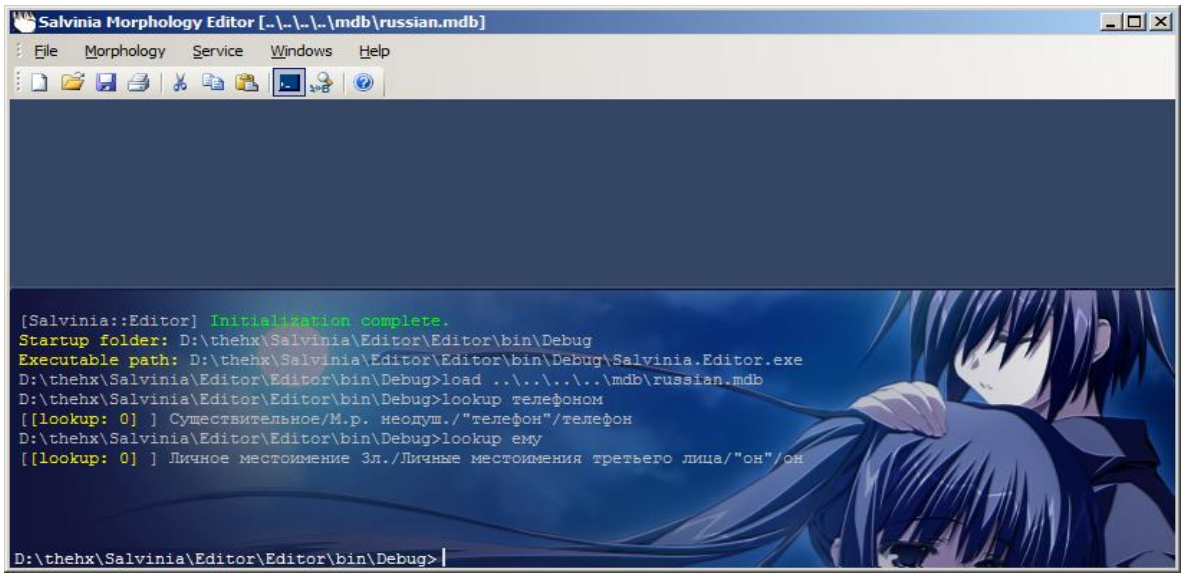

Рисунок 2. Поиск слов в базе знаний из служебной командной строки редактора БД

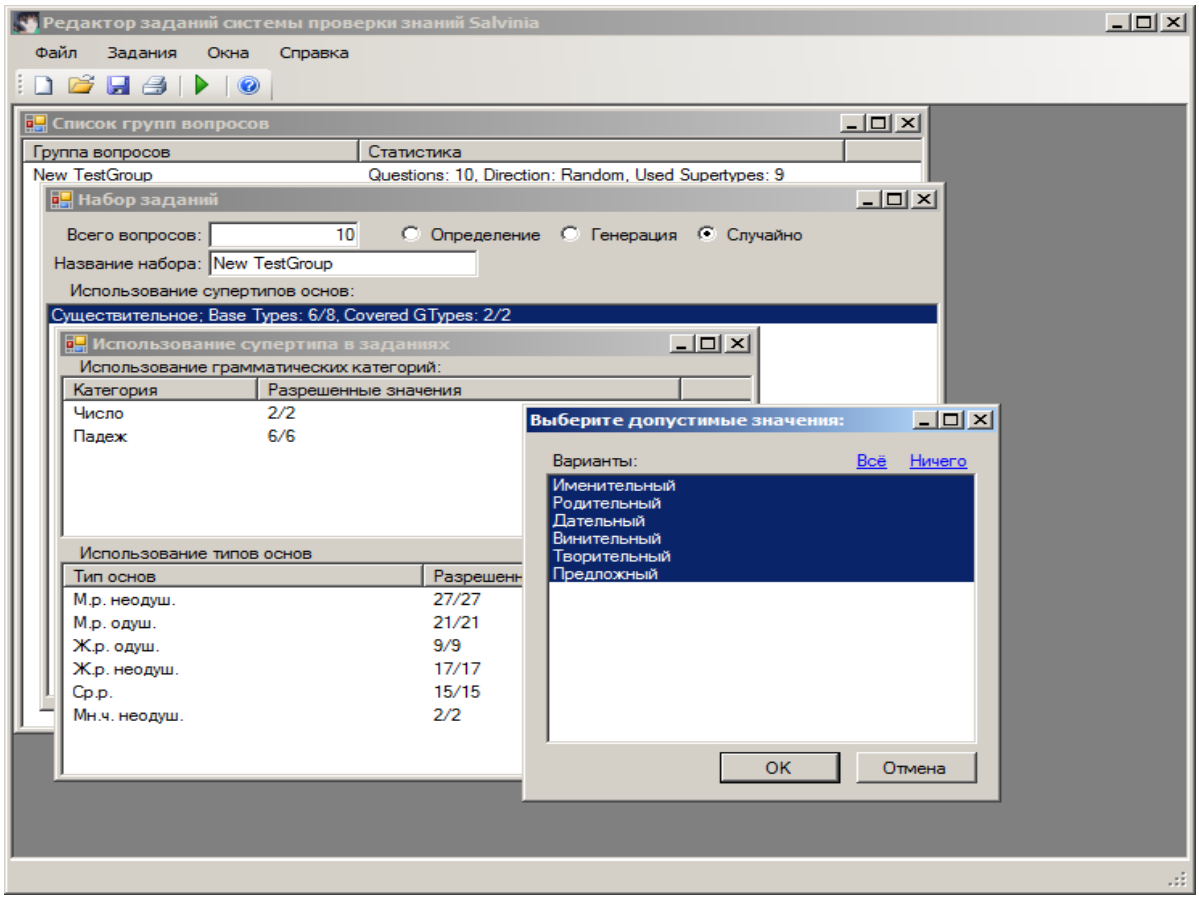

Рисунок 3. Редактирование состава тестовых заданий

- Salvinia Tester (рисунки 4, 5, 6, 7) – средство проверки знаний, работающее как в режиме обучения (с подсказками), так и в режиме

контроля, выполняющее проверку правильности ответов, ведущее подсчёт затраченного времени и составляющее полный протокол тестирования;

- Salvinia Log Viewer (рисунок 8) – средство просмотра протоколов тестирования.

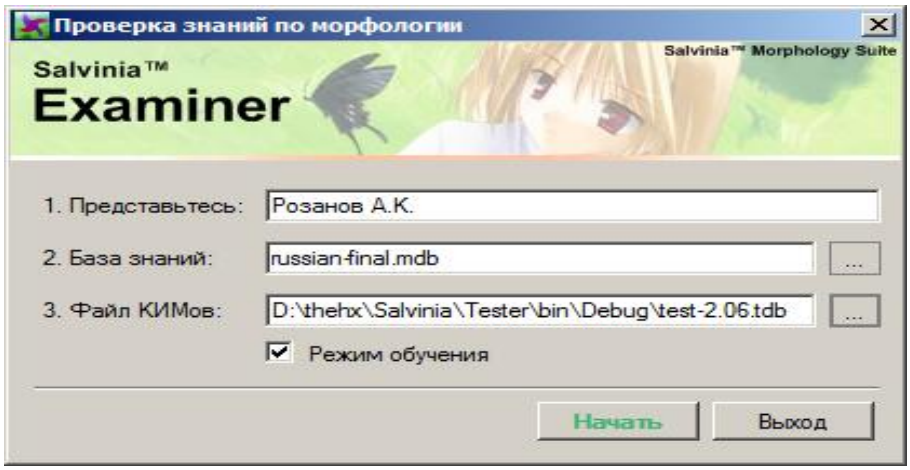

Рисунок 4. Вход в систему проверки знаний

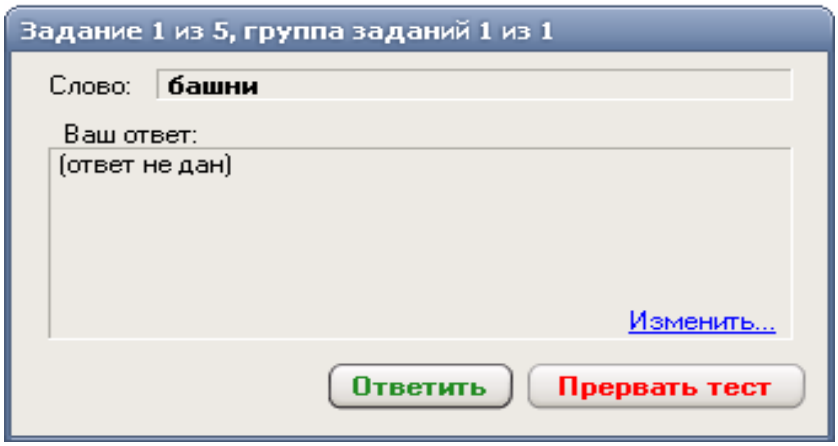

Рисунок 5.Задание на определение формы слова

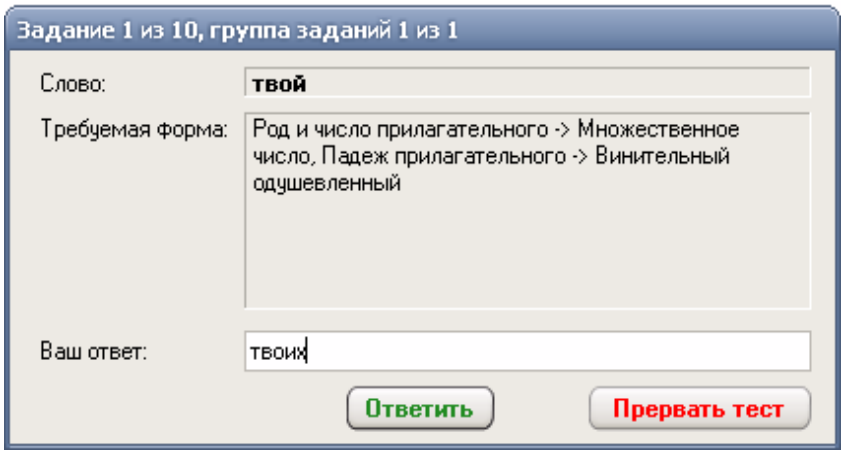

Рисунок 6. Задание на генерацию формы слова

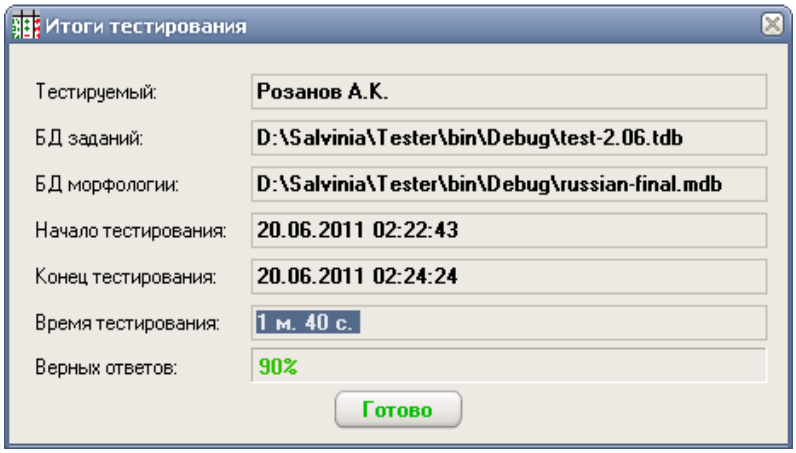

Рисунок 7.Результаты тестирования

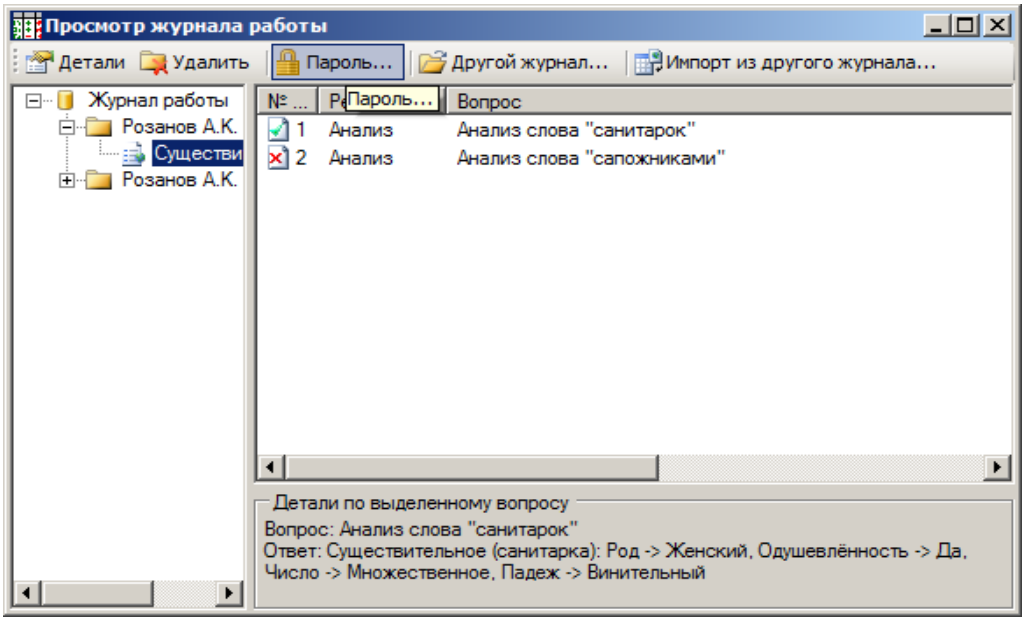

Рисунок 8. Протокол проведённого сеанса тестирования

3. Работа с системой

Последовательность действий при прохождении обучающимся теста:

1) ввод информации о тестируемом и подключение к базе знаний морфологии и базе заданий;
2) последовательный выбор или ввод ответов на предлагаемые вопросы;
3) внесение информации о тестировании в журнал событий и вывод результатов тестирования на экран.

Система контроля знаний работает в двух режимах:

- режим обучения (ответы анализируются, выводится справка и правильный ответ);
- режим контроля (выводится только окно статистики).

По окончании тестирования система переходит в изначальное состояние, и тестирование может быть начато заново.

Модуль тестирования открывает базу заданий и для каждой из групп запускает цикл на требуемое число повторений (указанное как «число вопросов» в группе заданий) генерации вопроса и проверки введенного или выбранного пользователем ответа. Логика проверки ответа приведена ниже.

1. В случае задачи определения формы слова указанная пользователем трактовка сравнивается со всеми допустимыми трактовками, которые получаются путём вызова необходимого метода модуля Salvinia Engine для активной базы морфологии, и если (и только если) пользователь ввёл подходящий вариант трактовки словоформы и этот вариант входит в подмножество, заданное текущей базой тестов, ответ пользователя считается верным.

2. В случае задачи генерации формы слова из данной основы слова всеми возможными способами (единственность способа необязательна, например, прилагательное «красный» имеет формы «красною» и «красной» для творительного падежа женского рода) получаются словоформы, имеющие требуемую грамматическую информацию. Если (и только если) ответ пользователя совпадает хотя бы с одной из полученных словоформ, он считается верным.

Пользователь в любой момент может остановить тестирование, нажав на соответствующую кнопку. Все вопросы, на которые ещё не был дан ответ, в этом случае не фиксируются в журнале событий. Вместе с тем они не считаются правильными, и доля верных ответов по-прежнему вычисляется как отношение данных верных ответов к запланированному общему числу вопросов.

Часть 2. ЭЛЕКТРОННАЯ ИНФОРМАЦИОННО-ПОИСКОВАЯ СИСТЕМА «РУССКО-АНГЛИЙСКИЙ МАТЕМАТИЧЕСКИЙ СЛОВАРЬ»

Введение

В настоящее время проблема информатизации образования и науки выдвинулась как приоритетная [1]. В этом контексте информатизации проблема качественного перевода становится всё более острой в связи с постоянным увеличением плотности информационных потоков и возрастающей потребностью в обмене информацией между людьми, говорящими на разных языках.

Английский язык занимает особое место в мире, науке, образовании и технике, он стал международным. Поэтому перевод научной работы на английский язык является одной из первоочередных задач любого автора.

Точность перевода имеет решающее значение при преобразовании с русского на английский язык сложного научного или технического текста. Зачастую знаний английской лексики, которыми владеет автор, становится недостаточно для точного воспроизведения законченной мысли на техническом английском языке, с использованием общепринятых терминов и привычных учёным рассматриваемой области науки языковых конструкций.

Поэтому возникает потребность в создании инструментария перевода на теоретической и прикладной базе проектирования и создания информационных систем.

Актуальность задачи подтверждается разработанными аналогами проекта [2-6]. Однако данные решения имеют закрытый код программы, что делает невозможным использование их для решения поставленной задачи.

Целью данной части является описание решения проблемы создания удобного для пользователя инструмента перевода. Таким инструментом явилась «Информационная система «Русско-английский словарь математических терминов»», разработанная авторами этой публикации. Система состоит из набора словарных статей, содержащих варианты перевода русских слов и словосочетаний на английский язык, и управляющей программы. Специфика словаря – именно математическая, так как в современной библиографии практически отсутствуют русско-английские математические словари общего содержания (без ориентации на конкретный раздел математики).

1. Русско-английский словарь математических терминов.

В основе информационной части системы лежит «бумажная» версия русско-английского словаря математических терминов, составленного А.И. Заволокиным и В.В. Мироновым. Словарь объемом 426 страниц содержит почти 40 000 слов, около 10 000 словарных статей и одобрен научно-методическим советом (НМС) по математике при Минобрнауки РФ [7]. Этот словарь не имеет прямых аналогов в России. В то же время составители опирались на его косвенные аналоги, то есть на англо-русские словари, а также на специализированные словари [8-14] (ставшие библиографической редкостью).

К другим достоинствам словаря по сравнению с косвенными аналогами можно отнести:

- более широкий охват специальных математических терминов и в особенности их словарных сочетаний;
- более точный перевод некоторых терминов в их сочетаниях с другими терминами;
- наличие терминов, появившихся в обороте сравнительно недавно.

2. Функциональное назначение, область применения, ограничения

В настоящее время проблема автоматизации труда переводчиков становится всё более острой в связи с постоянным увеличением плотности информационных потоков и возрастающей потребностью в обмене информацией между людьми, говорящими на разных языках.

Информационная система «Электронный русско-английский словарь математических терминов», как и «бумажный» аналог, разработана с целью облегчения работы по переводу научных текстов, она может служить подспорьем при написании статей на английском языке, подготовке презентаций, чтении специальной литературы, сокращая при этом время

подготовки. Система будет полезна иностранным студентам, обучающимся математике.

Разработанная информационная система представляет собой отдельное приложение для Windows требующее для своей работы только наличия платформы Microsoft .NET Framework 3.5, распространяемой бесплатно и доступной во всех версиях Windows, начиная с Windows XP. Программа не требует процедуры инсталляции и очень проста в обращении. Главное окно программы показано на рис. 1.

Разработанное приложение имеет интуитивно понятный интерфейс. Главное окно снабжено пользовательским меню, панелью инструментов и рабочей областью, разделённой на две части: область вывода списка статей словаря и область просмотра текста статей. Пользователь может выполнять поиск по словарю, вводя искомое слово или его часть в текстовое поле. На рис. 1 пользователь, введя «Аксио» в поле ввода искомого слова, ограничил набор статей до множества статей, заголовки которых содержат «Аксио».

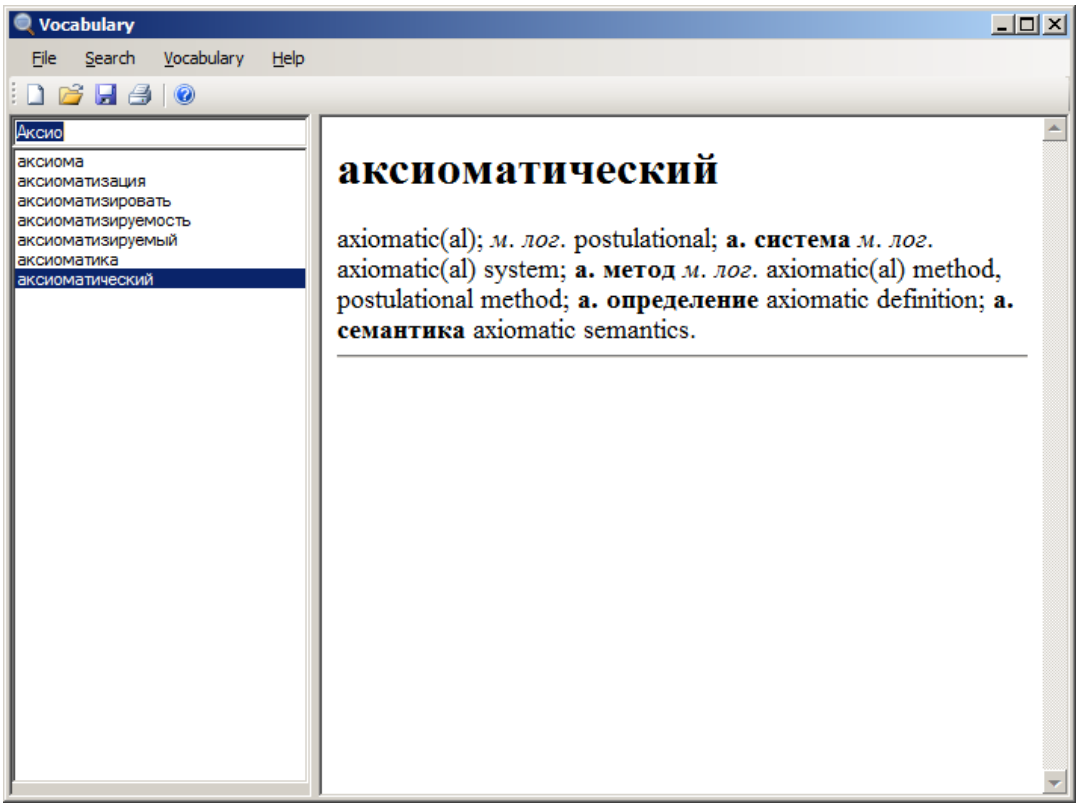

Рисунок 1. Главное окно информационной системы

Если пользователь начнёт вводить английские слова, система ограничит список статей, исключив те, в тексте которых введённые английские слова не встречаются.

Пользуясь пунктом меню “Search”, пользователь может ограничить результаты поиска по тексту статей только статьями, содержащими слова, начинающиеся с введённых символов (в интерфейсе приложения пункт меню “Strict Word Matching”, *рус.* «Строгое соответствие слов»), либо выводить также и слова, в которых введённые символы встречаются в середине.

Например, при поиске по слову rational в режиме нестрогого соответствия пользователь увидит как статью «рациональный», так и статью «иррациональный», в то время как при строгом поиске статья «иррациональный» будет исключена из результатов поиска.

Типичный сценарий работы пользователя с программой:

1) пользователь вводит (или начинает вводить) интересующее его слово, требующее перевода, в поле для поиска статей;
2) в списке найденных статей словаря, соответствующих критерию поиска, пользователь выбирает интересующее слово двойным щелчком мыши;
3) в окне просмотра текста статьи отображается содержимое статьи словаря, то есть допустимые варианты перевода слова.

Помимо этого, при включенном режиме отслеживания состояния буфера обмена Windows информационная система выдаёт оповещения о переводе слова, если пользователь помещает его в буфер обмена (копируя или вырезая текст в текстовом редакторе или интернет-браузере).

Пользователь может расширять имеющийся словарь, создавая новые статьи и изменяя существующие. Для этого в пользовательском меню есть пункт “Vocabulary”, содержащий команды для добавления, изменения и удаления статей словаря.

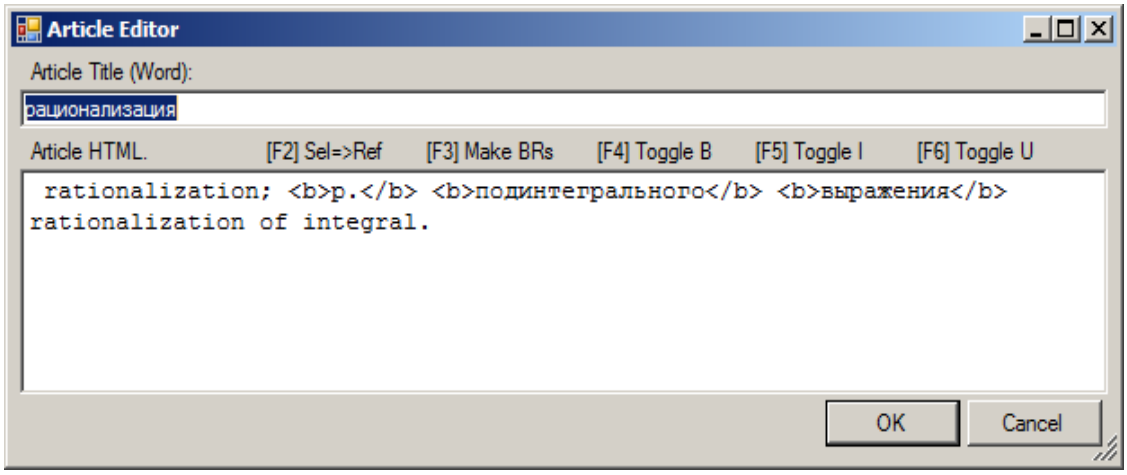

Рисунок 2. Окно редактора статей словаря

Редактор статей словаря (рис. 2) позволяет создавать статьи и применять элементарное форматирование к тексту. В частности, можно изменять параметры отображения шрифта (начертание, размер) и абзаца (отступы и выравнивание), а также оформлять ссылки на другие статьи в пределах файла словаря. Горячие клавиши (F2..F6) облегчают работу пользователя, позволяя «оборачивать» выделенные участки текста в соответствующий тег разметки, наделяя участок текста нужными свойствами – к примеру, делая выделенный текст полужирным или наклонным.

Внесённые в словарь изменения могут быть сохранены соответствующей командой “Save” меню “File” или при помощи одноимённой кнопки панели инструментов. Словарь хранится одним

файлом, имеющим расширение binvoc. Система при запуске может автоматически открывать последний из использовавшихся ранее файлов словарей.

Схема алгоритма генерации простой HTML-разметки на основе выделенного фрагмента документа Microsoft Word приведена на рис. 3.

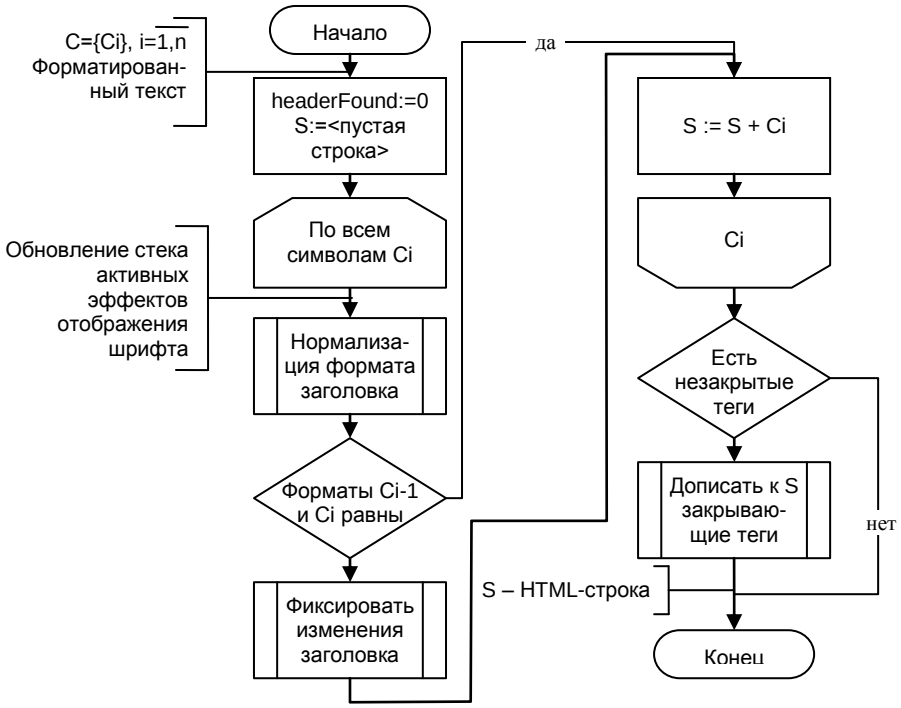

Рисунок 3. Укрупненная схема генерации HTML на основе текста статьи Word

В приведённой блок-схеме отсутствуют детали реализации (так, для выявления различий в оформлении и сохранения порядка закрытия тегов применяется стек открытых тегов; под нормализацией заголовка понимают правильное оформление заголовка статьи полужирным шрифтом без пробелов и знаков препинания в заголовке).

Под фиксацией изменений формата понимается такая последовательность закрывающих и открывающих тегов, отправляемая в формируемый документ, которая приводит к идентичному форматированию документа HTML; для этого записываются закрывающие теги для всех открытых тегов, включая вновь закрываемый, после чего открываются повторно теги, которые не следовало закрывать. При фиксации изменений, требующих открытия нового тега и не требующих закрытия уже открытых, дополнительные действия помимо открытия нового тега не требуются.

3. Структура и условия применения

Информационная система «Электронный словарь» имеет следующий файловый состав (таблица 1).

Исполняемый файл программы – это приложение Windows, которое следует запустить (например, выполнив двойной щелчок мышью по его значку в Проводнике), чтобы начать работу с информационной системой. Файл настроек программы хранит предпочтения пользователя (например, его выбор, открывать ли файл словаря сразу при запуске программы, использовать ли по умолчанию строгий поиск и так далее). Файл шаблона внешнего вида статьи задаёт оформление по умолчанию для всех статей в системе.

Таблица 1

Путь к файлу	**Описание назначения файла**
./Dictionary.exe	Исполняемый файл программы
./Dictionary.exe.config	Файл настроек программы
./Interface.htm	Файл шаблона стандартного внешнего вида статьи
./ru-en.binvoc	Файл русско-английского словаря (база статей)
./Include/*	Папка, содержащая служебные системные файлы

Пользователь, знающий язык HTML, может изменить этот файл, придав системе внешний вид по своему усмотрению, однако в принципе для корректной работы с системой модификация этого файла не требуется. Файл словаря (binvoc, от англ. binary vocabulary) – это файл, хранящий набор статей словаря, закодированный в двоичном формате, недоступном для открытия обычными текстовыми редакторами. В этом файле хранятся все полезные данные информационной системы. Вспомогательные файлы, хранящиеся в папке Include, используются при отображении статей словаря. Если пользователь принимает решение о модификации шаблона статей (Interface.htm), ему, скорее всего, потребуется хранить дополнительные ресурсы (например, изображения для использования в качестве заднего фона). Папка Include предусмотрена системой специально для этих целей – все файлы, содержащиеся в ней, используются для правильного отображения статей словаря в программе.

Пользователю для начала работы с программой требуется:

1) обеспечить нахождение в одной папке перечисленных выше файлов (Dictionary.exe, Dictionary.exe.config, Interface.htm, ru-en.binvoc) и папки Include со всем её содержимым (в изначальном варианте там присутствует лишь файл jquery-1.7.1.min.js) в одном и том же каталоге;

2) чтобы начать работать, запустить файл Dictionary.exe;

3) чтобы загрузить словарь в программу в первый раз, воспользоваться пунктом “Open Vocabulary” меню “File” и выбрать файл ru-en.binvoc в появившемся окне.

4. Используемые технические средства

Для корректной работы электронного словаря, а следовательно, и всей информационной системы, требуется наличие персонального компьютера под управлением ОС Windows XP или более новой, с установленной платформой Microsoft .NET Framework 3.5, бесплатно распространяемой на официальном сайте Microsoft. Подключение к сети для работы электронного словаря не требуется.

Требования к персональному компьютеру пользователя для работы с электронным словарём представлены в таблице 2.

Таблица 2

	Офисный	
	Минимум	Оптимум
Корпус (питание)	Mini/Midi Tower 350 Вт	Mini/Midi Tower 350 Вт
Материнская плата (чипсет)	AMD 740G + SB700 (CPU AMD) или Intel G41 + ICH7 (CPU Intel)	AMD 740G + SB710 (CPU AMD) или Intel G41 + ICH7 (CPU Intel)
ЦПУ	AMD Athlon II X2 240 или Intel Pentium Dual-Core E2180	AMD Athlon II X2 255 или Intel Pentium Dual-Core E5300
ОЗУ	2 GB DDR2 800 (2x1 GB)	2 GB DDR2 800 (2x1 GB)
Жесткий диск (HDD)	SATA 250 GB, 7200 об/мин	SATA 500 GB, 7200 об/мин
Сетевая и графическая платы	Интегрированные	Интегрированные

5. Пример работы с системой.

Допустим, необходимо перевести следующую теорему с русского языка на английский язык.

Теорема (В.В. Миронов). *Существуют такие бесконечно базируемые многообразия алгебр, объединение которых имеет уже конечный базис тождеств. Как следствие, такие многообразия не образуют полурешетки относительно операции объединения.*

Шаг 1. Выделим математические термины и ключевые слова, от точности перевода которых в наибольшей степени зависит корректность перевода теоремы: «*существуют*», «*бесконечно базируемые*», «*многообразие алгебр*», «*объединение*» (математическая операция), «*конечный*», «*базис», «тождество*».

Шаг 2. Выполним поиск первого из выделенных терминов в словаре (рис.4)

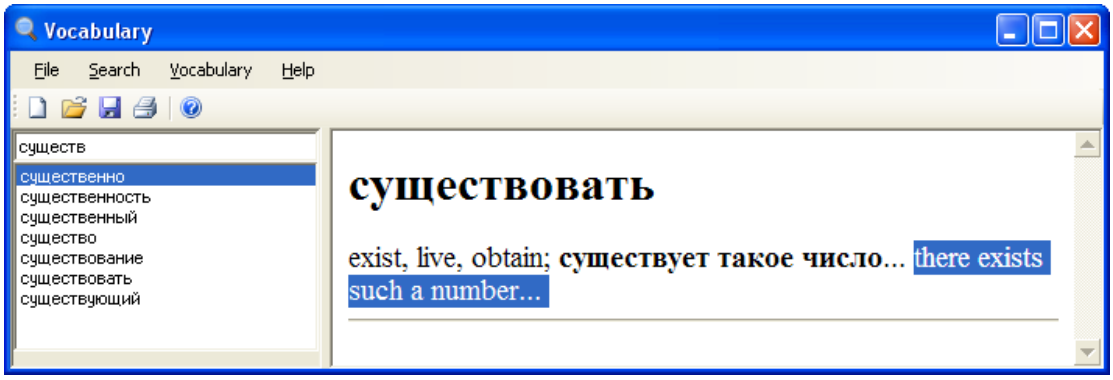

Рисунок 4. Поиск термина в словаре

Шаг 3. По аналогии найдём остальные термины и выпишем подходящие для перевода конструкции из соответствующих статей:

существует there exists, *бесконечно* infinitely, *базируемый* basable, *многообразие* manifold, *алгебра* algebra, *объединение* union, *конечный* finite, *базис* basis, *тождество* identity, *полурешетка* hemigrating, *операция* operation, *образовывать* produce, *следствие* consequence.

Шаг 4. Опираясь на правила грамматики английского языка, запишем вариант теоремы на английском языке.

Theorem (Mironov V.V.). *There exist such infinitely based algebra manifolds, the union of which already has a finite basis of identities. As a consequence, such manifolds do not form hemigratings under the union operation.*

Замечание. Перевод теоремы с помощью стандартного переводчика PROMT – X Translator GOLD на платформе Windows приводит к следующему результату.

Theorem. *Exist such indefinitely* ***базируемые*** *varieties of algebras which association has already final basis of identities. As a consequence, such varieties do not form* ***полурешетки*** *concerning operation of association.*

Перевод, осуществлённый с помощью онлайн-сервиса Google Translate, выглядит следующим образом.

Theorem. *There are infinitely based varieties of algebras whose union is already a finite basis. As a consequence, these varieties do not form a semilattice under the operation of association.*

Очевидно, что в обоих случаях качество перевода (несоответствия выделены в тексте) не может нас удовлетворить. *Конец замечания.*

Заключение

Авторами представлена информационная система «Электронный русско-английский словарь математических терминов», которая могла бы

войти (в той или иной степени) в национальную компьютерную сеть науки и высшей школы [15].

К достоинствам разработанной системы, кроме перечисленных ранее, можно отнести также следующие моменты:

- переносимость (нет необходимости установки, система не привязана к СУБД);
- небольшой объем (размер программы и словарей не превосходит 3 Мб);
- возможность импорта статей из документов Microsoft Word;
- двунаправленный перевод (русско-английский и англо-русский).

Электронные словари являются системами автоматической обработки текстов. Поэтому планируется расширить функциональные возможности разработанного словаря за счет обработки всех грамматических форм терминов с помощью универсального метода генерации и определения форм слов [16, 17].

Разработанная информационная система включает в себя средство для создания и редактирования статей словаря, следовательно, допускает, во-первых, дальнейшее развитие созданного словаря, во-вторых, создание новых словарей любого назначения.

Информационная система обладает достаточными возможностями для создания в перспективе не только словаря, но и математической энциклопедии.

Помимо этого, перспективной является интеграция созданной информационной системы с текстовыми редакторами и интернет-браузерами, с целью автоматизации перевода интересующих пользователя слов.

Справедливости ради следует отметить, что хотя большая часть слов, присутствующих в разработанной информационной системе, также присутствует в доступных в Интернете онлайн-словарях, энциклопедиях и системах машинного перевода (таких как dic.academic.ru, Yandex.Словари, Google Translate), объём и качество многих словарных статей разработанной системы (включая такие фундаментальные статьи, как *метод, область, группа, функция, коэффициент, схема* и многие другие) существенно превышают объём и качество соответствующих словарных статей в упомянутых выше Интернет-порталах. Так, для слова *метод* в разработанной информационной системе приводится 10 различных вариантов прямого перевода и 40 вариантов перевода слова *метод* с учётом контекста. Для сравнения, портал dic.academic.ru даёт 7 вариантов прямого перевода и три варианта перевода слова в контексте различных фраз.

Литература к главе 3

1. Болотов В.А. Основные направления информатизации общего образования //Информатизация общего образования. М.: Просвещение, 2003. С. 5-8.

2. Thiele D. Life cycle management using life cycle process standards. Веб-адрес статьи: http://www.fostas.ru/library/show_article.php?id=22

3. Clegg, Dai and Richard Barker. Case Method Fast-track: A RAD Approach Adison-Wesley, 1994.
4. Видеообзор электронного словаря AtomicDic, http://www.youtube.com/watch?v=7B7APF0DdJE
5. Проект Dicto, http://www.dicto.org.ru/
6. Описание электронного словаря GoldenDict: http://ru.wikipedia.org/wiki/GoldenDict
7. Заволокин А.И., Заволокина О.В., Миронов В.В. Математический русско-английский словарь. М.: Изд-во «Буки Веди», 2012. 426 с. ISBN 978-5-906069-89-4.
8. A.J. Lohwater's Russian-English dictionary of the mathematical sciences / R.P. Boas (ed.). American Mathematical Society, /1 ed. 1961, 2 ed. 1990.
9. Англо-русский словарь математических терминов /под ред. Александрова П.С. /1-е изд. М.: ИЛ, 1962, 2-е изд. М. :Мир, 1994.
10. Циммерман М., Веденеева К. Русско-английский научно-технический словарь переводчика. М.: Наука, 1993.
11. Коваленко Е.Г. Англо-русский математический словарь. Т. 1, 2. М.: Эрика, 1994.
12. Боровков К.А. Англо-русский, русско-английский словарь по теории вероятностей, статистике и комбинаторике. Москва-Филадельфия: SIAM, 1994.
13. Игнатьев-Каллэхэм Л. Русско-английский политехнический словарь. М.: Уайли, 1995.
14. Арушанян О.Б. Русско-английский словарь по прикладной математике и механике. М.: МГУ, 2003.
15. Куракин Д.В. Развитие национальной компьютерной сети науки и высшей школы // Педагогическая информатика. 2010. N 1. С.38-48.
16. Prutskov A.V. Algorithmic Provision of a Universal Method for Word-Form Generation and Recognition // Automatic Documentation and Mathematical Linguistics, 2011, Vol. 45. No. 5. P. 232-238.
17. Заволокин А.И., Миронов В.В. Алгоритмизация перевода при системном подходе к изучению английского языка// межвуз. сб. трудов «Математические методы в научных исследованиях». Рязань: Изд-во РГРТУ, 2012. С. 42-51.

Глава 4

ОПИСАНИЕ СИСТЕМ КОМПЬЮТЕРНОЙ АЛГЕБРЫ *MAPLE* И *WXMAXIMA*

1. Общие сведения о СКА, назначение, структура и функциональные возможности

Системы компьютерной алгебры (СКА, англ. Computer Algebra System, CAS) в настоящее время являются одними из необходимых компонентов информационных технологий, используемых в образовании. СКА могут применяться для решения широкого круга задач – от нахождения решения простейшего линейного уравнения в символьном виде до исследования серьезной научной работы, в которой результат желательно получить в аналитическом виде. Поэтому основной задачей любой СКА является представление результата в символьном (аналитическом) виде.

В процессе изучения дисциплины естественно-математического цикла одной из форм обучения является самостоятельное освоение студентами новых информационных технологий, в частности современных систем компьютерной математики (СКМ). Совместно с традиционными формами и методами обучения (лекции, упражнения, лабораторные работы) и в дополнение к основным современным языкам программирования (Pascal, C## и т.д.) студенты могут самостоятельно проводить анализ поставленной задачи в определенной СКМ.

Классификация СКА может проводиться по разным критериям [2, 4, 10, 12]. СКМ условно можно разделить на две категории:

- *системы для численных вычислений* (табличные процессоры MS Excel, система для статистических расчетов Statistica, системы для моделирования, анализа и принятия решения GPSS, AnyLogic);
- *системы компьютерной алгебры* (СКА).

СКА по функциональному назначению можно разделить на два типа (см. рис.1):

- *системы общего назначения* (Mathematica, Maple, MathCad, wxMaxima, MathLab и т.д.);
- *узкоспециализированные системы* [GAP (теория групп), Cadabra (тензорная алгебра), KANT (теория чисел), Calc3D (работа с 3D матрицами и векторами) и т.д.]

Основное назначение СКА – работа с числами и математическими выражениями (функциями, уравнениями, интегралами, матрицами и векторами, тензорами) в символьной форме. СКА работает по следующему правилу: математические объекты (матрицы, уравнения или их системы, основные алгебраические структуры и т.д.) задаются на входном языке в символьном виде. Интерпретатор СКА анализирует эти символьные объекты и переводит их во внутреннее представление. Затем символьный процессор, входящий в состав СКА, выполняет соответствующие алгоритмы внутренних

преобразований в рамках поставленной задачи и выдает ответ в символьном виде.

Алгоритмы внутренних представлений и преобразований в СКА имеют алгебраическую основу. СКА имеет так называемое синтаксическое дерево (систему списков) – внутреннее представление математического выражения в системе символьных вычислений. Суть символьных вычислений (аналитических преобразований) состоит в переписывании терма с помощью последовательного применения определенных правил из системы списков.

Рисунок 1. Классификация СКА по функциональному назначению

Современная СКА имеет в своем составе *ядро системы*, *интерфейсную оболочку (интерфейс пользователя)*, *библиотеки программных модулей и пакеты расширений* (см. рис. 2). Ядро системы реализовано на машинно-ориентированном языке, содержит операторы и функции для выполнения основных аналитических (символьных) преобразований математических выражений. В дополнение к ядру в СКА есть специализированные библиотеки программных модулей (пакеты). Эти библиотеки содержат систематизированные по назначению реализации алгоритмы обработки данных и решения конкретных типовых задач. Библиотеки программных модулей функционально расширяют ядро системы, а также обеспечивают возможности программирования на языке системы или на языке ее реализации.

Функциональные возможности современных СКА достаточно обширны [2, 4, 10]. Из основных возможностей отметим следующие:

1) *символьные и численные вычисления*:

- упрощение и факторизация аналитических выражений,
- подстановки выражений,
- интегрирование и дифференцирование в символьном виде,
- интегральные и дискретные преобразования (преобразования Фурье и Лапласа),
- интерполяция и экстраполяция;

2) *численное и символьное решение:*

- систем линейных и нелинейных алгебраических уравнений и неравенств,
- уравнений и систем обыкновенных дифференциальных уравнений,
- оптимизационных задач;

3) *решение задач линейной алгебры*:

- работа с основными объектами линейной алгебры (матрицами, векторами, тензорами),
- решение систем линейных алгебраических уравнений различными способами,
- формирование специальных матриц и работа с ними,
- работа с основными объектами линейных пространств, линейными операторами;

4) *возможности расширения системы*:

- создание процедур пользователя, задание внешних функций и процедур,
- применение подключаемых библиотек операторов и функций,
- применение пакетов расширения, специализированных на определенные классы вычислений,
- интеграция с другими математическими и иными системами,
- программирование на языках высокого уровня или на языке реализации системы.

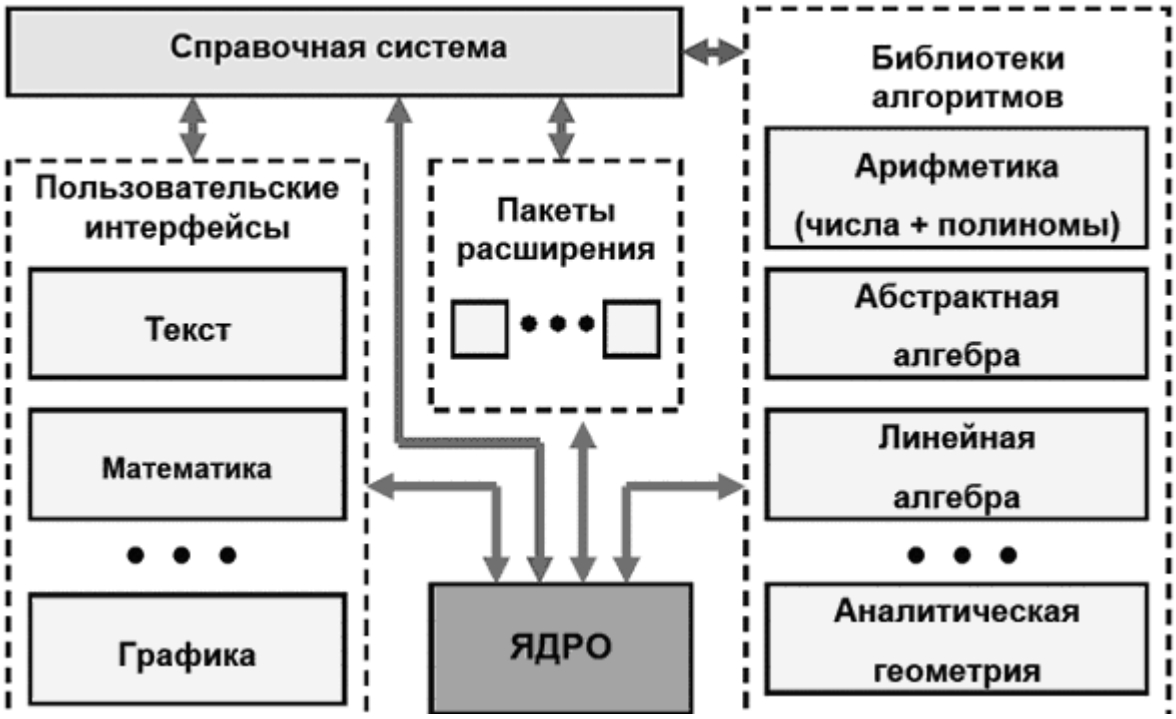

Рисунок 2. Структура современной СКА

В настоящее время лидерами среди коммерческих СКА являются *Mathematica* [7] и *Maple* [2, 3, 5, 13, 14]. Эти системы обладают мощными ядрами, развитыми интерфейсными оболочками, большим количеством программных модулей и пакетов расширений. Среди некоммерческих продуктов наиболее известными являются *wxMaxima* [8, 9], *Axiom*.

2. Обзор возможностей Maple в области решения задач линейной алгебры

Одной из самых популярных современных систем компьютерной алгебры в настоящее время является *Maple* [2, 3, 5, 13, 14]. В данном разделе кратко опишем основной инструментарий этой СКА в области решения задач линейной алгебры.

Начиная с версии 6, в *Maple* в дополнение к пакету *linalg* включен пакет *LinearAlgebra*, основанный на высокопроизводительных алгоритмах линейной алгебры. Это пакет библиотек программных модулей и функций фирмы *NAG* (*Numerical Algorithms Group*) – разработчик быстрых и высокоточных алгоритмов и программ. В состав пакета *LinearAlgebra* входит примерно 130 команд и процедур, предназначенных для решения соответствующих задач. Загрузка пакета осуществляется при помощи команды:

```
[> with(LinearAlgebra);
```

Все команды пакета *LinearAlgebra* можно условно разделить на следующие основные категории (см. рис. 3): *Constructors, Conversions, Data Structures, Eigenvalues, Functional Calculus, Queries, Solvers, Standard, Sub Operation, VectorSpaces.*

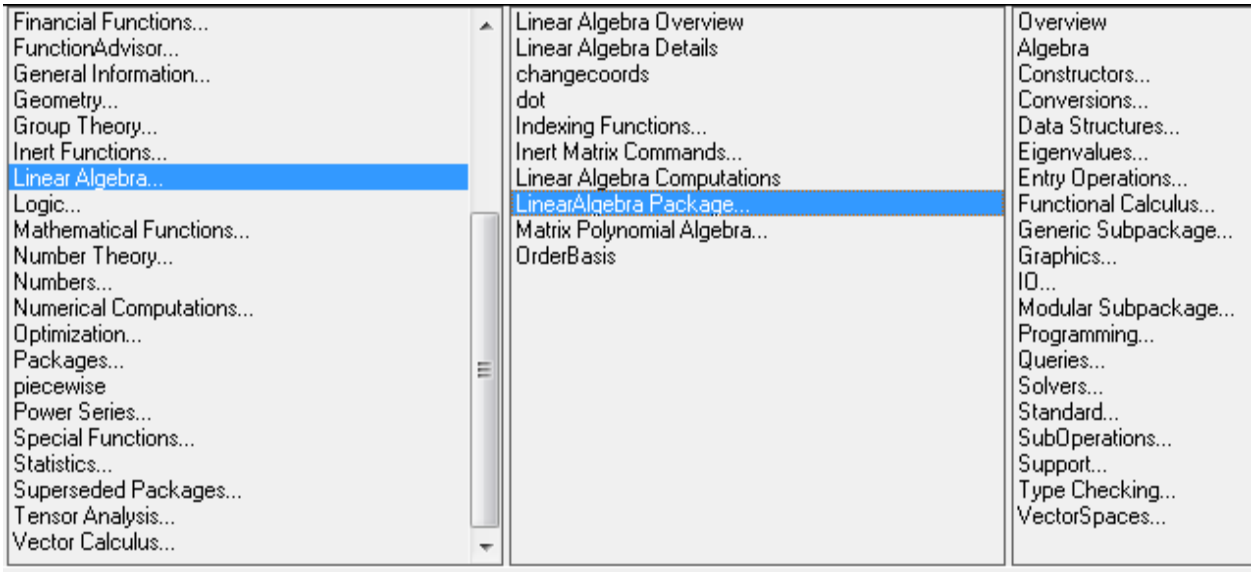

Рисунок 3. Основные категории пакета *LinearAlgebra СКА Maple*

Категория *Constructors* содержит процедуры конструирования специализированных видов матриц (см. табл. 1).

Таблица 1

Процедура	Назначение
`CharacteristicMatrix`	Конструирование характеристической матрицы по основной квадратной матрице
`ConstantMatrix`	Конструирование постоянной матрицы
`DiagonalMatrix`	Конструирование диагональной матрицы
`HankelMatrix`	Конструирует симметрическую квадратную матрицу
`IdentityMatrix`	Конструирует единичную матрицу
`JordanBlockMatrix`	Конструирует квадратную матрицу из жордановых блоков
`OuterProductMatrix`	Конструирует матрицу $A=\left(a_{ij}\right)$ размером $m\times n$ из произведения двух вектор-столбцов U, V по правилу $a_{ij}=u_i\cdot v_j$ $(i=\overline{1,m},\ j=\overline{1,n})$
`RandomMatrix`	Конструирует матрицу из случайных чисел
`SubMatrix`	Конструирует блок (подматрицу) из исходной матрицы
`SylvesterMatrix`	Конструирует матрицу Сильвестра
`UnitVector`	Конструирует единичный n-мерный арифметический вектор
`VandermondeMatrix`	Конструирует матрицу Вандермонда из элементов x_1, x_2, ..., x_n
`ZeroMatrix`	Конструирует нулевую матрицу

Процедуры категории *Conversions* служат для преобразования матричных данных из одного типа в другой.

Категория *Data Structures* ориентирована на конструирование в системе базовых структур матричной алгебры – матриц и векторов. Основные процедуры данной категории – процедуры `Matrix, Vector` предназначены для конструирования соответственно матриц и векторов с различными функциями. Процедуры `MatrixOptions` и `VectorOptions` выводят информацию соответственно о матрице или векторе.

Процедуры категории *Eigenvalues* предназначены для вычисления собственных значений, собственных векторов, сингулярных значений и минимальных многочленов квадратных матриц или матриц линейных операторов (см. табл. 2).

Таблица 2

Процедура	Назначение
`CharacteristicMatrix`	Составляет характеристическую матрицу для квадратной матрице
`CharacteristicPolynomial`	Конструирует для квадратной матрицы ее характеристический многочлен
`Eigenvalues`	Вычисляет собственные значения квадратной матрицы
`Eigenvectors`	Вычисляет собственные векторы квадратной матрицы
`MinimalPolynomial`	Вычисляет минимальный многочлен для квадратной матрицы
`SingularValues`	Вычисляет сингулярные значения квадратной матрицы

Так, например, процедура `Eigenvectors` имеет следующие наиболее часто используемые форматы вызова:

```
lambda, Matrix_Eigenvectors:=Eigenvectors(A)
Eigenvectors(A, output='list')
```

В первом случае процедура выдает вектор-столбец собственных значений `lambda` и матрицу `Matrix_Eigenvectors`, столбцами которой являются соответствующие собственные векторы матрицы `A`.

Во втором случае при использовании параметра `output='list'` процедура выдает список, каждый элемент которого имеет вид

`[lambda, k, {eigenvectors}]`

где `lambda – собственное значение, k` – его алгебраическая кратность, `eigenvectors – соответствующий собственный вектор (если` $k>1$, то выдается список из линейно независимых собственных векторов`)`.

Ниже приводится фрагмент программы, в которой вычисляются собственные векторы матрицы.

```
[> restart; with(LinearAlgebra):
[> A:=Matrix(4,4,[[3,-6,9,0], [1,-2,3,0], [-3,6,-9,0],[0,0,0,5]]):
```

```
[> lambda,Matrix_Eigenvectors:=Eigenvectors(A);
   Eigenvectors(A, output='list');
```

$$\lambda, Matrix\_Eigenvectors := \begin{bmatrix} 5 \\ -8 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & -1 & -3 & 2 \\ 0 & \frac{-1}{3} & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\left[\left[-8, 1, \left\{\begin{bmatrix} -1 \\ \frac{-1}{3} \\ 1 \\ 0 \end{bmatrix}\right\}\right], \left[5, 1, \left\{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}\right\}\right], \left[0, 2, \left\{\begin{bmatrix} -3 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}\right\}\right]\right]$$

Категория *Functional Calculus* включает процедуры для работы с математическими функциями, аргументами которых являются матрицы. Процедура `MatrixFunction` имеет в качестве своих аргументов определенную пользователем функцию (аналитическое выражение), квадратную матрицу и независимую скалярную переменную (аргумент функции). Результатом работы процедуры является значение функции, аргументом которого является квадратная матрица.

Категория *Queries* содержит ряд логических процедур для изучения внутренней структуры матриц (см. табл. 3).

Таблица 3

Процедура	Назначение
`ColumnDimension`	Определяет количество столбцов в матрице (векторе)
`RowDimension`	Определяет количество строк в матрице (векторе)
`Dimension`	Определяет размерность (количество строк и столбцов) в матрице или векторе
`Equal`	Логическая процедура. Проверка на равенство двух матриц или двух векторов
`IsDefinite`	Логическая процедура. Проверяет квадратную матрицу на знакоопределенность
`IsOrthogonal`	Логическая процедура. Проверяет квадратную матрицу на ортогональность
`IsSimilar`	Логическая процедура. Проверяет две квадратные матрицы на подобие
`Rank`	Вычисляет ранг матрицы

Процедуры категории *Solvers* предназначены для исследования на совместность систем линейных алгебраических уравнений и нахождения их решений различными способами (см. табл. 4).

Таблица 4

Процедура	Назначение
`BackwardSubstitute`	Процедура нахождения решения системы линейных алгебраических уравнений с верхнетреугольной основной матрицей этой системы
`BidiagonalForm`	Конструирует матрицу бидиагональной формы для квадратной матрицы A - матрицу B, так что $A = U \cdot B \cdot Vt$, где U, Vt – ортогональные матрицы
`FrobeniusForm`	Приводит квадратную матрицу A к матрице F Фробениуса (к рациональной канонической форме): $F = Q^{-1} \cdot A \cdot Q$, где столбцами матрицы Q (матрицы соответствующего преобразования) являются векторы канонического рационального базиса
`GaussianElimination`	Производит элементарные преобразования Гаусса над ненулевой матрицей A, приводя ее к ступенчатому (верхнетреугольному) виду
`GenerateEquations`	Конструирует систему линейных алгебраических уравнений в координатной форме записи из коэффициентов заданной матрицы и вектор-столбца свободных коэффициентов со списком неизвестных системы
`HermiteForm`	Вычисляет эрмитову нормальную форму матрицы из квадратной матрицы
`JordanForm`	Конструирует матрицу $J = Q^{-1} \cdot A \cdot Q$ жордановой формы для квадратной матрицы A, где Q – матрица соответствующего преобразования
`LeastSquares`	Процедура нахождения решения системы линейных алгебраических уравнений или матричного уравнения методом наименьших квадратов. Если рассматривается несовместная система линейных алгебраических уравнений вида $Ax = b$ с основной матрицей A и вектор-столбцом b, то функция `LeastSquares` выдает вектор-столбец x наилучшего приближения к решению, найденный методом наименьших квадратов (евклидова норма $\|Ax - b\|$ минимальна)
`LinearSolve`	Процедура нахождения решения системы линейных алгебраических уравнений, записанной в матричном виде $A \cdot x = b$. В качестве необязательных параметров выбираются метод решения и имена для свободных переменных,

	Если система неопределенна
`LUDecomposition`	Процедура разложения ненулевой матрицы $\underset{m\times n}{A}$ в виде произведения $\underset{m\times n}{A} = \underset{m\times m}{L}\cdot\underset{m\times n}{U}$ нижнетреугольной матрицы $\underset{m\times m}{L}$ и верхнетреугольной матрицы $\underset{m\times n}{U}$ (*LU-разложение*, или *LU-факторизация* матрицы)
`QRDecomposition`	Процедура QR-факторизации, то есть представления ненулевой матрицы A в виде произведения $A = Q\cdot R$, где Q – матрица, столбцами которой являются ортонормированные векторы (в случае, если матрица Q – квадратная, то она есть ортогональная матрица), R – верхняя треугольная матрица, причем диагональные элементы матрицы R положительны
`ReducedRowEchelonForm`	Производит элементарные преобразования Гаусса-Жордана над матрицей, приводя ее к верхнетреугольному и нижнетреугольному виду $\begin{pmatrix} 1 & 0 & \dots & 0 & \alpha_{1,r+1} & \dots & \alpha_{1n} \\ 0 & 1 & \dots & 0 & \alpha_{2,r+1} & \dots & \alpha_{21} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & \alpha_{r,r+1} & \dots & \alpha_{rn} \end{pmatrix}$
`SmithForm`	Приводит квадратную многочленную матрицу $A(\lambda)=\left(a_{ij}(\lambda)\right)$ n-го порядка к матрице Смита канонической формы: $S(\lambda)=diag\left(\mu_1(\lambda),\ \mu_2(\lambda),\ \dots,\ \mu_r(\lambda)\right)$, $r=rangA(\lambda)$; $\mu_1(\lambda),\ \mu_2(\lambda),\ \dots,\ \mu_r(\lambda)$ – инвариантные многочлены (инвариантные множители) матрицы $A(\lambda)$
`TridiagonalForm`	Конструирует матрицу T тридиагональной формы для квадратной симметрической матрицы $A=\left(a_{ij}\right)$ n-го порядка, то есть такую матрицу, что $T = Q\cdot A\cdot Q^T$, где Q – ортогональная матрица соответствующего преобразования

Процедуры категории *Standard* определяют стандартные команды матричной алгебры (см. табл. 5).

Таблица 5

Процедура	Назначение
`Add`	Вычисление линейных комбинаций, составленных из матриц (векторов) с числовыми коэффициентами
`Adjoint`	Процедура вычисления присоединенной матрицы для заданной квадратной матрицы
`BilinearForm`	Конструирует билинейную форму $F(v_1, v_2) = v_1^T \cdot A \cdot v_2$ из двух вектор-столбцов v_1, v_2 одного размера относительно квадратной матрицы A n-го порядка
`CrossProduct`	Вычисляет векторное произведение в координатной форме двух геометрических векторов
`Determinant`	Вычисляет определитель квадратной матрицы
`DotProduct`	Вычисляет стандартное скалярное произведение двух векторов одинаковой размерности
`KroneckerProduct`	Конструирует тензор (матрицу) Кронекера произведения двух матриц
`MatrixAdd`	Вычисление линейной комбинации из двух матриц одинаковых размеров
`MatrixInverse`	Процедура вычисления обратной матрицы для неособенной квадратной матрицы. Если же матрица особенная или не квадратная, то при использовании дополнительного параметра процедура выдает псевдообратную матрицу
`MatrixMatrixMultiply`	Процедура произведения двух матриц
`MatrixNorm, Norm, VectorNorm`	Процедуры для нахождения норм матриц (векторов) различными способами
`MatrixScalarMultiply`	Процедура произведения матрицы на скаляр
`VectorScalarMultiply`	Процедура произведения вектора на скаляр
`MatrixVectorMultiply`	Процедура произведения матрицы на вектор-столбец
`Minor`	Вычисляет минор M_{ij} для элемента a_{ij} в квадратной матрице $A = (a_{ij})$ n-го порядка или выписывает подматрицу (блок матрицы) из прямоугольной матрицы путем удаления соответствующих строк и столбцов (при использовании дополнительного параметра)
`Normalize`	Процедура нормализации вектора, приведение его к единичному вектору (по выбранной норме)

Trace	Вычисляет след квадратной матрицы
Transpose	Операция транспонирования матрицы
VectorAdd	Вычисление линейных комбинаций из двух векторов одинаковых размеров
VectorMatrixMultiply	Операция произведения вектор-строки на матрицу

Категория *Sub Operation* содержит процедуры, ориентированные на изменение внутренней структуры матрицы (проведение преобразований над строками и столбцами, см. табл. 6).

Таблица 6

Процедура	Назначение
Column	Возвращает по номерам столбцов элементы этих столбцов в прямоугольной матрице
ColumnOperation	Производит элементарные преобразования над столбцами матрицы
RowOperation	Производит элементарные преобразования над строками матрицы
DeleteColumn	Возвращает матрицу, полученную в результате удаления из исходной матрицы перечисленных столбцов
DeleteRow	Возвращает матрицу, полученную в результате удаления из исходной матрицы перечисленных строк
Pivot	Обнуляет элементы фиксированного столбца в прямоугольной матрице относительно ведущего ненулевого элемента
Row	Возвращает по номерам строк элементы этих строк в прямоугольной матрице

Процедуры категории *VectorSpaces* предназначены для работы с конечномерными линейными векторными пространствами (см. табл. 7).

Таблица 7

Процедура	Назначение
Basis	Определяет базис векторного пространства, заданного множеством векторов
ColumnSpace	Определяет базис векторного пространства, составленного из вектор-столбцов матрицы
GramSchmidt	Проводит по заданной системе векторов процесс ортогонализации Грама - Шмидта, возвращает ортогональную или ортонормированную систему векторов
IntersectionBasis	Определяет базис пересечения двух и более

	векторных пространств, каждое из которых задается в виде списка или множества векторов
`NullSpace`	Определяет базис для ядра матрицы (нуль-пространства, пространства решений однородной системы линейных алгебраических уравнений)
`RowSpace`	Определяет базис векторного пространства, порожденного строками матрицы
`SumBasis`	Определяет базис суммы двух и более векторных пространств, каждое из которых задается в виде списка или множества векторов

Категория *Generic SubPackage* содержит общие реализации алгоритмов линейной алгебры над полями и кольцами. Большинство процедур данной категории аналогичны соответствующим процедурам пакета *LinearAlgebra* с тем лишь отличием, что результат зависит от конкретной реализации поля или кольца. Вот перечень некоторых процедур категории: `CharacteristicPolynomial` (характеристический многочлен), `Determinant` (определитель), `GaussianElimination` (элементарные преобразования Гаусса - Жордана), `LinearSolve` (решение СЛАУ), `MatrixInverse` (обратная матрица), `MatrixMatrixMultiply` (произведение двух матриц), `MatrixVectorMultiply` (произведение матрицы на вектор), `NullSpace` (нахождение нуль-пространства матрицы), `ReducedRowEchelonForm`, `RREF`(приведение матрицы к ступенчатому виду), `SmithForm` (нахождение канонической формы матрицы).

Еще одна категория *Modular SubPackage* содержит набор процедур для выполнения матричных вычислений в кольцах целых чисел по модулю. Набор процедур данной категории схож с предыдущими категориями.

3. Обзор возможностей wxMaxima в области решения задач линейной алгебры

Основной отличительной чертой современного состояния информационных технологий является то, что коммерческие программные продукты во многих случаях полностью или частично можно заменить свободным (некоммерческим) программным продуктом с открытым исходным кодом. В большинстве случаев эти программные комплексы публикуются и распространяются под лицензией GNU GPL (универсальная общественная лицензия, General Public License). Это означает, что они могут свободно запускаться и копироваться; можно изучать исходные коды программ, модифицировать эти программы, внося изменения и создавая свои собственные пакеты расширений и программные модули.

Maxima [8, 9] – свободная универсальная система компьютерной алгебры, потомок системы *Macsyma. Macsyma*, созданная в 60 - х годах 20 - го века в Массачусетском технологическом институте (MIT) по заданию министерства энергетики США (DOE), являлась родоначальником всех систем компьютерной математики и первой системой аналитических

вычислений. Система создавалась на языке искусственного интеллекта Lisp и изначально была закрытым программным продуктом. Университетская разработка, несмотря на ее фундаментальное значение, не смогла выдержать конкуренции с ведущими коммерческими СКА *Maple* и *Mathematica*. Благодаря профессору Техасского университета В. Шелтеру, который получил разрешение от DOE на открытую публикацию кода программы под лицензией GNU GPL, была создана СКА *Maxima*.

Как и любая другая СКА, *Maxima* имеет в своем составе ядро системы, интерпретатор макроязыка (написанный на *Lisp*), библиотеки программных модулей и множество пакетов расширений (packages), написанных на макроязыке системы. Являясь универсальной системой компьютерной алгебры, *Maxima* обладает большими возможностями при решении практически всех разделов математического и функционального анализов, линейной и тензорной алгебр, теории чисел и групп, статистики.

Maxima свободно распространяется по лицензии GNU GPL под любую из следующих операционных систем: *Linux, MAC, Windows.* Для каждой из этих операционных систем существуют две реализации системы – консольная *Maxima* и система с графическим интерфейсом пользователя (GUI) – *wxMaxima* (графическая надстройка над *Maxima*). Последняя доступная версия *wxMaxima* на момент написания имеет порядковый номер 16.12.0.

Рассмотрим СКА *Maxima* с GUI *wxMaxima*. При запуске *wxMaxima* появляется окно программы (на рис. 4 оно дано открытой панелью Математика).

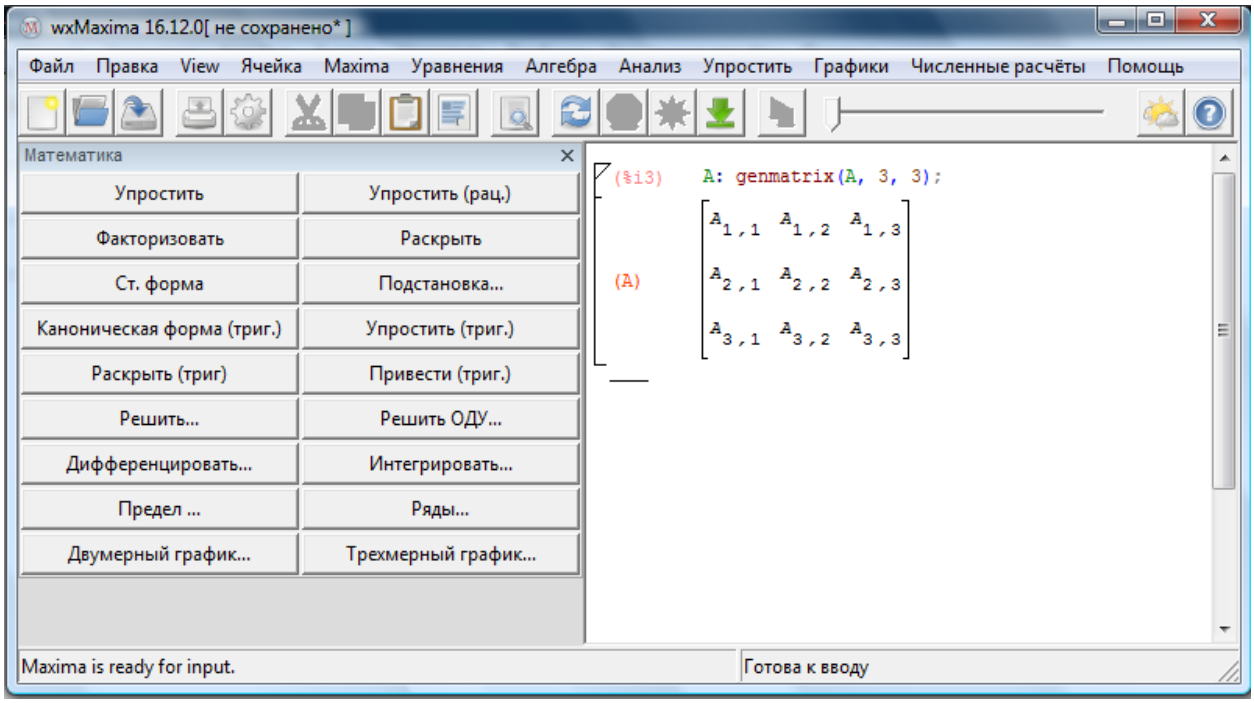

Рисунок 4. Окно программы *wxMaxima*

Ввод командной строки завершается нажатием комбинаций клавиш Ctrl / Enter. В следующем листинге представлен пример применения *wxMaxima* для

нахождения обратной матрицы третьего порядка и приведения матрицы к ступенчатому виду.

```
(i%3) A: matrix([a,2,3],[-2,4,6],[-3,6,19]); invert(A);
      triangularize(A);
```

$$(\%o2)\quad \begin{bmatrix} \frac{40}{40\,a+40} & -\frac{20}{40\,a+40} & 0 \\ \frac{20}{40\,a+40} & \frac{9+19\,a}{40\,a+40} & \frac{-6-6\,a}{40\,a+40} \\ 0 & \frac{-6-6\,a}{40\,a+40} & \frac{4+4\,a}{40\,a+40} \end{bmatrix}$$

$$(\%o3)\quad \begin{bmatrix} -2 & 4 & 6 \\ 0 & -4\,a-4 & -6\,a-6 \\ 0 & 0 & -40\,a-40 \end{bmatrix}$$

Типы данных в *wxMaxima* весьма разнообразны: числа разного формата, строковые данные, константы, переменные, массивы, векторы и матрицы. Система *wxMaxima* имеет полный набор встроенных основных и специальных математических функций. Кроме того, есть множество системных функций и функций из пакетов расширения системы. Каждый пакет перед работой требует загрузки с помощью команды:

```
load("Имя_пакета расширения");
```

Рассмотрим подробнее возможности *wxMaxima* в плане решения задач линейной алгебры. Для этого в *wxMaxima* есть большое количество процедур и функций. Их можно условно разделить по категориям и специализированным пакетам: *Matrices* (категория, в которой содержатся базовые процедуры, функции и переменные линейной алгебры), *Vectors* (категория, процедуры и функции которой ориентированы на задачи векторного анализа), *Package eigen* (пакет процедур для вычисления собственных значений и собственных векторов), *Package diag* (пакет процедур для построения нормальных форм матриц, жордановых форм), *Linear algebra* (*linearalgebra*, пакет процедур для решения задач, связанных с линейными и евклидовыми пространствами), *Linear equations* (категория, в которой собраны процедуры для исследования и решения систем линейных алгебраических уравнений различными методами).

Категория *Matrices* СКА *wxMaxima* содержит базовые процедуры, функции и переменные, предназначенные для решения задач линейной алгебры:

- `addcol(A, c1, …, cn)` – добавляет к матрице *A* справа столбцы *c1, …, cn* (в виде списков или матриц) согласованной размерности;
- `addrow(A, r1, …, rn)` – добавляет к матрице *A* снизу строки *r1, …, rn* (в виде списков или матриц) согласованной размерности;
- `adjoint(A)` – вычисляет присоединенную матрицу к квадратной матрице *A*;

- augcoefmatrix(*[eqn_1, …, eqn_m], [x_1, …, x_n]*) – возвращает расширенную матрицу из коэффициентов при переменных *x_1, …, x_n* и коэффициентов свободных членов из системы линейных уравнений *eqn_1, …, eqn_m*;
- charpoly(*A, lambda*) – конструирует характеристический многочлен для квадратной матрицы *A* по переменной *lambda* [при использовании команды ratsimp(charpoly(B,x))] будет выведен упрощенный вид характеристического многочлена);
- coefmatrix(*[eqn_1, …, eqn_m], [x_1, …, x_n]*) – возвращает основную матрицу из коэффициентов при переменных *x_1,…,x_n* из системы линейных уравнений *eqn_1, …, eqn_m*;
- col(*A, i*) – возвращает *i*-столбец матрицы *A*;
- columnvector(*L*) – возвращает вектор-столбец из элементов списка *L*;
- determinant(A) – вычисляет определитель квадратной матрицы *A*;
- diagmatrix(n, x) – конструирует диагональную матрицу *n*-го порядка с диагональными элементами *x*;
- echelon(A) – преобразует матрицу A к ступенчатому виду, используя элементарные преобразования (при этом на главной диагонали ступенчатой матрицы будут находиться единицы);
- ematrix(m, n, x, i, j) – конструирует матрицу размером *m*x*n*, все элементы которой равны 0, за исключением элемента в позиции *i, j*, который равен *x*;
- entermatrix(m, n) – создание матрицы размером в интерактивном режиме;
- genmatrix(A,m,n) – конструирует матрицу A размера *m*x*n*, каждый элемент которой определяется выражением, зависящим от индексов;
- ident(n) – конструирует единичную матрицу порядка *n*;
- invert(A) – вычисляет обратную матрицу для квадратной матрицы A. Вычисление обратной матрицы проводится при помощи метода *LU*-декомпозиции. В случае, если матрица является вырожденной, выдается сообщение об ошибке вида

```
expt: undefined: 0 to a negative exponent.
-- an error. To debug this try: debugmode(true);
```

- list_matrix_entries(A) – возвращает список элементов матрицы A;
- matrix(row_1, …, row_n) – конструирует прямоугольную матрицу со строками row_1, …, row_n. Каждая строка задается в виде списка элементов одной длины;
- matrixmap(f, A) – вычисляет матрицу f(A) для матрицы A, где f есть наперед заданная функция;
- matrix_element_add, matrix_element_mult, matrix_element_transpose – опции переменных, позволяющие переопределять стандартные бинарные операции над матрицами (сложение, умножение, транспонирование) новыми операциями;

- `mattrace(A)` – возвращает след (сумму элементов на главной диагонали квадратной матрицы) матрицы `A`;
- `minor(A,i,j)` – вычисляет минор для элемента квадратной матрицы `A`, находящийся на пересечении i-й строки, j-го столбца;
- `rank(A)` - вычисляет ранг матрицы `A`;
- `row(A,i)` – возвращает `i`-ю строку матрицы `A`;
- `setelmx(`*x*`,i,j,A)` – заменяет в матрице `A` элемент, находящийся в i-й строке ,j-го столбца, на выражение *x*;
- `submatrix` – процедура, конструирующая подматрицу определенного размера из матрицы `A`. Имеет следующие форматы записи:

```
submatrix(i_1, …, i_m, A, j_1, …, j_n)
submatrix(i_1, …, i_m, A)
submatrix(A, j_1, …, j_n)
```

- `transpose(A)` – процедура построения транспонированной матрицы к матрице `A`;
- `zeromatrix(m,n)` – процедура построения нулевой матрицы размером `mxn`.

Процедуры и функции категории *Vectors* служат для решения задач векторного анализа (нахождение градиента, оператора Лапласа, различных произведений векторов).

Пакет *Package eigen* содержит процедуры и функции для вычисления собственных значений и собственных векторов квадратных матриц. Загрузка происходит при помощи команды

```
load("eigen");
```

Рассмотрим некоторые процедуры и функции пакета *Package eigen:*

- `innerproduct(x,y)` – вычисляет стандартное скалярное (внутреннее) произведение векторов `x,y` (сумма произведений соответствующих компонент);
- `unitvector(x)` – возвращает нормированный (единичный по евклидовой норме) вектор для заданного вектора `x`;
- `columnvector(x)` – преобразовывает вектор-строку x (список) в вектор-столбец;
- `gramschmidt(`*x*`,`*F*`)` – проводит процесс ортогонализации Грама – Шмидта системы векторов *x* относительно скалярного произведения, определенного функцией *F* . При отсутствии параметра *F* проводит процесс ортогонализации относительно стандартного скалярного произведения

```
(%i53)  x:matrix([1,2,2],[2,1,-2],[1,2,-1])$
        y:gramschmidt(sys);
        map(innerproduct,[y[1],y[2],y[3]],[y[2],y[3],y[1]]);
(y)     [[1,2,2],[2,1,-2],[-2/3,2/3,-1/3]]
(%o53)  [0,0,0];
```

• eigenvalues(*A*) – возвращает собственные значения матрицы *A*. Результат выдает в виде списка [[s1,s2, …,sn],[k1,k2,…,kn]] собственных значений s1,s2, …,sn с указанием их соответствующих алгебраических кратностей k1,k2,…,kn;

• eigenvectors(*A*) – вычисляет собственные векторы матрицы *A*. Результат виден в виде списка из двух элементов. Первый элемент – список [[s1,s2, …,sn],[k1,k2,…,kn]] собственных значений s1,s2, …,sn с указанием их соответствующих алгебраических кратностей k1,k2,…, kn, второй элемент – список соответствующих собственных векторов матрицы *A*.

```
(%i2)  load("eigen")$
       kill(all)$
       A:matrix([2,2,-2],[2,5,-4],[-2,-4,5]);
       eigenvectors(A);
```

$$(A)\quad \begin{bmatrix} 2 & 2 & -2 \\ 2 & 5 & -4 \\ -2 & -4 & 5 \end{bmatrix}$$

(%o2) $[[[10,1],[1,2]],[[[1,2,-2]],[[1,0,\frac{1}{2}],[0,1,1]]]]$

• uniteigenvectors(*A*) – вычисляет нормированные (единичные по евклидовой норме) собственные векторы матрицы *A*. Результат выдается в виде списка из двух элементов. Первый элемент – список [[s1,s2, …,sn],[k1,k2,…,kn]] собственных значений s1,s2, …,sn с указанием их алгебраических кратностей k1,k2,…,kn, второй элемент – список соответствующих нормированных собственных векторов матрицы *A*.

Пакет *Package diag* – пакет процедур для построения нормальных форм матриц, жордановых форм. Рассмотрим список процедур данного пакета.

• diag([s1,s2,…,sn]) – конструирует блочную матрицу из списка s1,s2,…,sn матриц или отдельных чисел;

• JF(lambda,n) – конструирует жорданову клетку n-го порядка с собственным значением lambda;

• jordan(A) – возвращает жорданову форму матрицы A в виде определенного списка. Каждый элемент списка также является списком из двух элементов. Первым элементом является собственное значение матрицы A. Вторым элементом является натуральное число, являющееся длиной жорданова блока для данного собственного значения (эти числа перечислены в порядке убывания);

• dispJordan(I) – возвращает матрицу жордановой канонической формы, соответствующую списку I (список I должен быть получен командой jordan(A));

- `minimalPoly(I)` – возвращает минимальный многочлен матрицы жордановой формы, соответствующей списку `I` (список `I` должен быть получен командой `jordan(A)`);
- `ModeMatrix(A,[jordan_info])` – возвращает неособенную матрицу B такую, что матрица $B^{-1}AB = JF$ есть матрица жордановой канонической формы, соответствующая матрице `A`;
- `mat_function(f,A)` – вычисляет значение функции `f(A)`, где `f` есть аналитическое выражение, `A` есть матрица.

Пакет *Linear algebra* содержит процедуры для решения задач линейной алгебры, связанных с линейными и евклидовыми пространствами. Перечислим некоторые из этих процедур:

- `addmatrices(f, M1, …, Mn)` – вычисляет сумму матриц `M1, …, Mn` одинаковых размеров с использованием специальной аддитивной функции `f`;
- `columnop(A,i,j,lambda)` – функция, возвращающая матрицу, полученную в результате умножения `j`-го столбца матрицы A на число lambda и сложения с `i`-м столбцом матрицы A;
- `columnswap(A,i,j)` – возвращает матрицу, полученную в результате смены местами столбцов с номерами `i, j` матрицы A;
- `columnspace(A)` – возвращает линейную оболочку в виде

`span(v1,...,vn),`

где вектор-столбцы `v1,...,vn` есть базис пространства столбцов матрицы `A`

```
(%i3)   M:matrix([1,2,3,0],[3,4,7,-2],[4,6,10,-2])$
        columnspace(M);
```

$$(\%o3)\quad \mathrm{span}\left(\begin{pmatrix}1\\3\\4\end{pmatrix},\begin{pmatrix}2\\4\\6\end{pmatrix}\right);$$

- `cholesky(A,field)` – возвращает разложение Холецкого матрицы `A`. При использовании параметра `field` проводит разложение над заданным полем;
- `ctranspose(A)` – возвращает комплексно-сопряженную матрицу для матрицы A (если матрица A вещественная, то результатом является соответствующая транспонированная матрица);
- `diag_matrix(s1,s2,…,sn)` – конструирует блочную диагональную матрицу из списка `s1,s2,…,sn` матриц. Получается матрица, на главной диагонали которой располагаются матрицы `s1,s2,…,sn`, а вне диагонали - соответствующего размера нулевые матрицы. Если в качестве параметров процедуры используются числа, то выдается соответствующая диагональная квадратная матрица;
- `dotproduct(v1,v2)` – вычисляет стандартное скалярное произведение двух вектор-столбцов `v1,v2`;

- `eigens_by_jacobi(A)` – вычисляет собственные значения и собственные векторы симметрической матрицы `A` по методу вращения (методу Якоби). При использовании процедуры в виде `eigens_by_jacobi(A, field_type)` в качестве второго аргумента используется специально определенное поле `field_type`. В результат выполнения процедуры выдается список из двух элементов. Первый список - собственные значения матрицы `A`, второй – соответствующие собственные векторы;
- `lu_factor(A)` – функция возвращает LU-разложение квадратной матрицы `A`. Результат выдается в виде списка из трех элементов;
- `get_lu_factors(x)` – возвращает список вида `[P,L,U]`, где `x` – результат выполнения процедуры *x*=`lu_factor(`*A*`)`, L есть нижнетреугольная матрица с единицами на главной диагонали, U есть верхнетреугольная матрица с единицами на главной диагонали, причем выполняется равенство $A = P\,L\,U$. Ниже приведен пример LU-разложения квадратной матрицы третьего порядка:

```
(%i1)   kill(all)$ A:matrix([1,2,0],[2,0,2],[1,1,1])$

(%i15)  x:lu_factor(A);
        y:get_lu_factors(x);
        P:y[1]$ L:y[2]$ U: y[3]$ P.L.U;
```

$$(x)\quad \left[\begin{pmatrix}1 & 2 & 0\\ 2 & -4 & 2\\ 1 & \frac{1}{4} & \frac{1}{2}\end{pmatrix}, [1,2,3], generalring\right]$$

$$(y)\quad \left[\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}, \begin{pmatrix}1 & 0 & 0\\ 2 & 1 & 0\\ 1 & \frac{1}{4} & 1\end{pmatrix}, \begin{pmatrix}1 & 2 & 0\\ 0 & -4 & 2\\ 0 & 0 & \frac{1}{2}\end{pmatrix}\right]$$

$$(\%o15)\quad \begin{pmatrix}1 & 2 & 0\\ 2 & 0 & 2\\ 1 & 1 & 1\end{pmatrix}$$

- `hessian(f(x1,x2,…,xn),[x1,x2,…,xn])` – возвращает матрицу Гессе (квадратную матрицу из частных производных второго порядка) для данной функции `f(x1,x2,…,xn)` переменных `x1,x2,…,xn`;
- `hilbert_matrix(n)` – возвращает матрицу Гильберта порядка `n`;
- `identfor(A)` – возвращает для квадратной матрицы A *n*-го порядка соответствующую единичную матрицу *n*-го порядка;
- `invert_by_lu(A)` – вычисляет матрицу, обратную к матрице `A`, используя метод LU-разложения;

- jacobian([f1,f2,…,fm],[x1,x2,…,xn]) – возвращает матрицу Якоби (матрицу из частных производных первого порядка) для списка функций f1,f2,…,fm от переменных x1,x2,…,xn;
- kronecker_product(A,B) – возвращает произведение Кронекера двух матриц A,B;
- lu_backsub(M,b) – функция, возвращающая решение уравнения $A \cdot x = b$, где M есть результат функции lu_factor(A), b есть вектор-столбец соответствующего размера;
- mat_norm(A, p) – возвращает p-норму матрицы A (в качестве параметра p могут использоваться значения 1, inf, frobenius);
- matrix_size(A) – возвращает список из двух чисел – количества строк и столбцов матрицы A;
- mat_unblocker(A) – разблокирует матрицу A, если первоначально матрица была создана как блочная матрица;
- nullspace(A) – возвращает линейную оболочку в виде
span(v1,...,vn),
где система вектор-столбцов v1,...,vn есть базис пространства строк (нуль-пространство, ядро) матрицы A;
- nullity(A) – возвращает размерность базиса пространства строк (нуль-пространства) матрицы A;
- orthogonal_complement(v1, …, vn) – возвращает линейную оболочку в виде

span(u1,...,um),

где система вектор-столбцов u1,...,um есть базис ортогонального дополнения к подпространству, порожденному вектор-столбцами

```
(%i82) v1:matrix([1],[2],[1])$ v2:matrix([2],[4],[0])$
       v3:matrix([3],[6],[1])$
       Basis:orthogonal_complement(v1,v2,v3);
```

(Basis) $\mathrm{span}\left(\begin{bmatrix} 4 \\ -2 \\ 0 \end{bmatrix}\right)$;

- rank(A) – определяет ранг матрицы A;
- rowop(A,i,j,lambda) – процедура, возвращающая матрицу, полученную в результате умножения j-й строки матрицы A на число lambda и сложения с i-й строкой матрицы A;
- rowswap(A,i,j) – возвращает матрицу, полученную в результате смены местами строк с номерами i, j матрицы A;
- vandermonde_matrix([s1,s2,…,sn]) – возвращает матрицу Вандермонда, соответствующую списку чисел s1,s2,…,sn;
- zerofor(A) – возвращает нулевую матрицу соответствующего размера для матрицы A.

Пакет *Linear equations* предназначен для исследования и решения СЛАУ различными методами:

- `dgesv(A, b)` – процедура, позволяющая найти решение матричного уравнения вида *Ax=b* с квадратной матрицей `A`, где `b` есть согласованная с `A` матрица. Решение основано на использовании метода LU-разложения матрицы `A`;
- `globalsolve` – логическая глобальная переменная, используемая совместно с процедурой `linsolve` для решения системы линейных алгебраических уравнений. По умолчанию значение переменной равно `false`, что означает, что после нахождения решения системы уравнений процедурой `linsolve` теряются значения найденных переменных (обратиться к ним невозможно).

```
(%i1)   kill(all)$
        globalsolve:false;
(globalsolve)  false
```

```
(%i11)  S:solve([x[1]+2*x[2]+x[3]=2, 2*x[1]-x[2]+x[3]=-5],
        [x[1], x[2],x[3]]);
        [x[1], x[2], x[3]];
(S)     [[x1 = -(3 %r3+8)/5, x2 = -(%r3-9)/5, x3 = %r3]]
(%o11)  [x1, x2, x3]
```

После команды присвоения `globalsolve:true` становится возможным напрямую обращение к найденным значениям переменным:

```
(%i1)   kill(all)$ globalsolve:true$
```

```
(%i3)   S:solve([x[1]+2*x[2]+x[3]=2, 2*x[1]-x[2]+x[3]=-5],
        [x[1], x[2],x[3]]);
        [x[1], x[2], x[3]];
(S)     [[x1 : -(3 %r4+8)/5, x2 : -(%r4-9)/5, x3 : %r4]]
(%o3)   [-(3 %r4+8)/5, -(%r4-9)/5, %r4] ;
```

- `linsolve([expr_1, …, expr_m], [x_1, …, x_n])` – основная процедура для нахождения решения системы линейных уравнений, определяемая списком *expr_1, …, expr_m* относительно списка переменных *x_1, …, x_n*. Если значение переменной `globalsolve` есть `true`, то после нахождения решения можно напрямую обратиться к значению каждой переменной;
- `linsolve_params` – логическая переменная. Если ее значение равно `true`, то процедура `linsolve` генерирует символы вида %r, использующиеся для подстановки в свободные и базисные переменные в случае неопределенности системы линейных уравнений. Если значение переменной `linsolve_params`

равно `false`, то решение системы выдается в виде выражений базисных переменных через свободные. Если значение переменной `linsolve_params` равно `true`, то, зная общее решение системы, можно найти частное решение системы;

- `linsolvewarn` – логическая переменная. Если ее значение равно `true`, то в случае зависимых уравнений в системе пишется сообщение вида

`"Dependent equations eliminated"`

(зависимые уравнения устранены) с номерами устранимых уравнений

```
(%i2)    kill(all)$ linsolvewarn:true$
         solve([3*x[1]+2*x[2]+x[3]=2,
         -x[1]+3*x[2]=4,2*x[1]+5*x[2]+x[3]=6], [x[1],x[2],x[3]]);
solve: dependent equations eliminated: (3)
(%o2)    [[x1 : -(3 %r11+2)/11, x2 : -(%r11-14)/11, x3 : %r11]]  ;
```

- `triangularize(A)` – преобразует матрицу `A` к верхнетреугольному (ступенчатому) виду, используя элементарные преобразования.

Литература к главе 4

1. Аладьев В.З. Компьютерная алгебра // Альфа, № 1:Гродно, ГрГУ, 2001.
2. Аладьев В.З. Системы компьютерной алгебры: Maple: Искусство программирования. М.: Лаборатория базовых знаний, 2006. 792 с.
3. Аладьев В.З. Программирование и разработка приложений в Maple: монография / В.З. Аладьев, В.К. Бойко, Е.А. Ровба. Гродно: ГрГУ; Таллинн: Межд. акад. ноосферы, Балт. отд. 2007. 458 с.
4. Дьяконов В.П. Энциклопедия компьютерной алгебры. М.: ДМК-Пресс, 2009. 1264 с.
5. Дьяконов В.П. Maple 10/11/12/13/14 в математических расчетах. М.: ДМК-Пресс, 2011. 800 с.
6. Дэвенпорт Дж., Сирэ И., Турнье Э. Компьютерная алгебра. М.: Мир, 1991. 352 с.
7. Половко А. М. Mathematica для студентов. - СПб.: БХВ-С-Петербург, 2007.
8. Система компьютерной алгебры Maxima [Электронный ресурс]. Режим доступа: http://maxima.sourceforge.net/ru/.
9. Стахин Н.А. Основы работы с системой аналитических (символьных) вычислений Maxima (программное обеспечение для решения задач аналитических (символьных) вычислений): учеб. пособие. М.,: 2008. 86 с.
10. Таранчук В.Б. Основные функции систем компьютерной алгебры. Минск: БГУ, 2013.59 с.
11. Aladjev V., Bogdevicius M., Vaganov V. Systems of Computer Algebra: A New Software Toolbox for Maple // Int. Conf. on Software Engin. Res. and Practice, 2004.
12. List of computer algebra systems [Электронный ресурс]. Режим доступа: http://en.wikipedia.org/wiki/List_of_computer_algebra_system.
13. Maplesoft. Books [Электронный ресур]. Режим доступа: http://maplesoft.com/books/index.aspx.
14. Maple Product History [Электронный ресурс]. Режим доступа: http://maplesoft.com/product/maple/history/.

Глава 5

ПРИМЕНЕНИЕ СИСТЕМ КОМПЬЮТЕРНОЙ АЛГЕБРЫ *MAPLE* И *WXMAXIMA* ДЛЯ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ

1. Линейные пространства, подпространства. Линейные операторы

Перечислим основные виды задач, возникающие при изучении разделов “Линейные пространства”, “Линейные операторы”:

- исследование на линейную зависимость системы векторов,
- построение базиса конечномерного линейного пространства,
- различные способы задания (описания) арифметических подпространств,
- нахождение базиса в сумме и пересечении двух (или более) линейных подпространств,
- построение алгебраического дополнения подпространства,
- нахождение инвариантных и корневых подпространств относительно линейного оператора,
- разложение линейного пространства в сумму корневых подпространств,
- задача диагонализации линейного оператора.

В дополнение к имеющимся в пакете LinearAlgebra СКА Maple процедур пакет `VectorSpaces` содержит ряд процедур и функций для решения перечисленых выше задач. Загрузка пакета производится командой

```
> read "D:VectorSpaces.m": with(VectorSpaces);
```

[*BasisColumnSpace* , *DependenceMatrix* , *DependencePolynom* , *DependenceVector* , *InternalExternal* , *Invariant_SubSpace* , *SummIntersectBasis*]

Перечислим процедуры и функции пакета `VectorSpaces` (см. табл. 1).

Таблица 1

Название процедуры	Описание, предназначение процедуры
`DependencePolynom`	Процедура исследования на линейную зависимость системы многочленов
`DependenceMatrix`	Процедура исследования на линейную зависимость системы матриц
`DependenceVector`	Процедура исследования на линейную зависимость системы вектор-столбцов
`BasisColumnSpace`	Процедура построения базиса линейного пространства $\mathbf{Lin}\left(\overline{a_1}, \overline{a_2}, \ldots, \overline{a_k}\right)$, порожденного системой вектор-столбцов $\overline{a_1}, \overline{a_2}, \ldots, \overline{a_k}$
`InternalExternal`	Процедура перехода от внутреннего способа описания подпространства к внешнему способу

	задания подпространства
SummIntersectBasis	Процедура построения базиса суммы и пересечения двух подпространств, заданных внутренним способом (в виде линейной оболочки порождающих вектор-столбцов)
Invariant_SubSpace	Процедура нахождения инвариантных (корневых) подпространств относительно линейного оператора, заданного матрицей в базисе вещественного линейного пространства

Исследование на линейную зависимость систем векторов в конечномерном линейном векторном пространстве

Одной из самых распространенных задач линейной алгебры является задача исследования на линейную зависимость систем векторов в конечномерном линейном векторном пространстве. В качестве элементов (объектов) линейного пространства здесь могут быть взяты системы многочленов, матриц, вектор-столбцов (вектор-строк), линейных форм, то есть такие системы векторов в конечномерных линейных пространствах, которые подлежат арифметизации.

Как известно, общий подход для исследования на линейную зависимость системы векторов $S=\{a_1,\ a_2,\ ...,\ a_k\}$ конечномерного линейного пространства V состоит в том, что составляется равенство

$$\sum_{i=1}^{k}\lambda_i a_i \equiv \lambda_1 a_1+\lambda_2 a_2+...+\lambda_k a_k=\theta \qquad (5.1)$$

(θ – нулевой вектор линейного пространства V) с весовыми коэффициентами $\lambda_1,\ \lambda_2,\ ...,\ \lambda_k$ из поля P. При этом система S является линейно зависимой, если существуют числа $\lambda_1,\ \lambda_2,\ ...,\ \lambda_k\in P$, среди которых есть хотя бы одно отличное от нуля, при которых выполняется равенство (5.1).

Рассмотрим на примере пространства $P_n[t]$ (многочленов относительно переменной t степеней, не превышающей n) алгоритм исследования на линейную зависимость. Пусть дана система многочленов:

$$S=\{p_1(t),\ p_2(t),\ ...,\ p_k(t)\},\ p_i(t)=a_n^{(i)}t^n+a_{n-1}^{(i)}t^{n-1}+...+a_1^{(i)}t+a_0^{(i)}\ (i=\overline{1,\ k}). \qquad (5.2)$$

Для исследования на линейную зависимость системы (5.2) составляем равенство для определения весовых коэффициентов $\lambda_1,\ \lambda_2,\ ...,\ \lambda_k$:

$$\lambda_1 p_1(t)+\lambda_2 p_2(t)+...+\lambda_k p_k(t)=\theta(t)\ \ (\theta(t)=0t^n+0t^{n-1}+...+0t+0).$$

Записав последнее равенство в координатной форме, получим

$$\lambda_1\left(a_n^{(1)}t^n+a_{n-1}^{(1)}t^{n-1}+...+a_1^{(1)}t+a_0^{(1)}\right)+\lambda_2\left(a_n^{(2)}t^n+a_{n-1}^{(2)}t^{n-1}+...+a_1^{(2)}t+a_0^{(2)}\right)+...$$

$$+\lambda_k\left(a_n^{(k)}t^n+a_{n-1}^{(k)}t^{n-1}+...+a_1^{(k)}t+a_0^{(k)}\right)=0t^n+0t^{n-1}+...+0t+0.$$

Приравнивая коэффициенты при одинаковых степенях переменной t в левой и правой частях этого равенства, получаем ОСЛАУ для определения коэффициентов λ_1, λ_2, ..., λ_k разложения:

$$\begin{cases} a_n^{(1)}\lambda_1 + a_n^{(2)}\lambda_2 + \ldots + a_n^{(k)}\lambda_k = 0, \\ a_{n-1}^{(1)}\lambda_1 + a_{n-1}^{(2)}\lambda_2 + \ldots + a_{n-1}^{(k)}\lambda_k = 0, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_0^{(1)}\lambda_1 + a_0^{(2)}\lambda_2 + \ldots + a_0^{(k)}\lambda_k = 0. \end{cases} \qquad (5.3)$$

Обозначив через $\underset{(n+1)\times k}{A} = \left(a_i^{(j)}\right)$ $(i = \overline{0,\ n},\ j = \overline{1,\ k})$ матрицу системы (5.3) размером $(n+1)\times k$, а через $\Lambda = \left(\lambda_1 \quad \lambda_2 \quad \ldots \quad \lambda_k\right)^T$ – вектор-столбец весовых коэффициентов, систему (5.3) можно записать в матричной форме

$$A \cdot \Lambda = \bar{0}_{n+1}. \qquad (5.4)$$

Известно, что если $rang(A) = r < k$, то система (5.4) имеет ненулевые решения относительно $\Lambda = \left(\lambda_1 \quad \lambda_2 \quad \ldots \quad \lambda_k\right)^T$, что означает линейную зависимость системы $S = \{a_1,\ a_2,\ \ldots,\ a_k\}$.

Процедура `DependencePolynom` пакета `VectorSpaces` исследует на линейную зависимость систему многочленов $S = \{p_1(t),\ \ldots,\ p_k(t)\}$. В случае линейной зависимости процедура находит подсистему S', являющуюся максимальной линейно независимой системой, и выражает остальные многочлены системы S через многочлены системы S' в виде линейной комбинации.

```
> restart; with(LinearAlgebra):
> read "D:VectorSpace.m"; with(VectorSpaces):
> n:=3: k:=5:
>a[1,3]:=-1: a[1,2]:=1: a[1,1]:=-1: a[1,0]:=1:
 a[2,3]:=-2: a[2,2]:=3: a[2,1]:=4: a[2,0]:=0:
 a[3,3]:=1: a[3,2]:=-2: a[3,1]:=-4: a[3,0]:=2:
 a[4,3]:=-3: a[4,2]:=3: a[4,1]:=-2: a[4,0]:=4:
 a[5,3]:=-7: a[5,2]:=9: a[5,1]:=5: a[5,0]:=3:
>for i from 1 by 1 to k do
  p[i]:=a[i,n]*t^n:
  for j from n-1 by -1 to 0 do
    p[i]:=p[i]+a[i,j]*t^(j):
  end do:
end do:
> # формирование исходной системы многочленов
Sys_Polynom:=[p[1], p[2], p[3], p[4], p[5]];
```

$$Sys\_Polynom := [-t^3 + t^2 - t + 1,\ -2t^3 + 3t^2 + 4t,\ t^3 - 2t^2 - 4t + 2,\ -3t^3 + 3t^2 - 2t + 4,\ -7t^3 + 9t^2 + 5t + 3]$$

```
> DependencePolynom(Sys_Polynom);
```

$$\Lambda := \begin{bmatrix} 2\,c_3 - 3\,c_5 \\ c_3 - 2\,c_5 \\ c_3 \\ -c_3 \\ c_5 \end{bmatrix}$$

$$p_4 := 2\,p_1 + p_2 + p_3$$

$$p_5 := 3\,p_1 + 2\,p_2$$

В результате применения процедуры выводятся решения системы (5.4) (вектор-столбец Λ) и равенства, выражающие зависимость многочленов, не являющихся базисными, через базисные многочлены.

Построение базиса в конечномерном линейном пространстве

Для каждого конечномерного линейного пространства можно построить базис. Рассмотрим алгоритм построения базиса в линейном пространстве $\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$, порожденного системой $S = \left\{\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right\}$ вектор-столбцов.

Составим матрицу A размером $n \times k$, столбцами которой являются векторы $\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}$. Приведем ее к ступенчатому виду A' с помощью элементарных преобразований над строками [можно допустить и перестановку местами столбцов, если при этом запоминать номера (индексы) столбцов]. Столбцы ступенчатой матрицы A', соответствующие столбцам ее базисного минора, определяют порождающие базисные вектор-столбцы в $\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$.

Ранг $r = rang\left(A\right)$ определяет ранг системы S векторов (максимальное количество линейно независимых векторов системы).

Соберем базисные векторы в фундаментальную матрицу F размером $n \times r$:

$$F = \left(\overline{b_1}\quad \overline{b_2}\quad ...\quad \overline{b_r}\right),\ \overline{b_i} \in \mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)\ \left(i = \overline{1,\ r}\right),$$

где $\mathbf{B} = \left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_r}\right)$ – базис пространства $\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$ (максимальная порождающая система линейно независимых векторов). При этом размерность $\dim\left(\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)\right) = r$.

Если требуется найти координатный вектор-столбец $\overline{\alpha} = \left(\alpha_1\quad \alpha_2\quad ...\quad \alpha_r\right)^T$ вектора $\overline{a} \in \mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$ в базисе $\mathbf{B}$, то необходимо решить систему

$$\underset{n\times r}{F} \cdot \underset{r\times 1}{\overline{\alpha}} = \underset{n\times 1}{\overline{a}},$$

которая имеет единственное решение относительно α_i $\left(i = \overline{1,\ r}\right)$.

В частности, для определения координатных вектор-столбцов в базисе $\mathbf{B}=\left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_r}\right)$ векторов $\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}$ необходимо решить систему

$$\underset{n\times r}{F}\cdot\underset{r\times k}{\Lambda}=\underset{n\times k}{A},$$

где $\underset{r\times k}{\Lambda}=\left(\lambda_{ij}\right)$ – матрица, столбцами которой являются соответствующие координатные вектор-столбцы векторов $\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}$ в базисе $\mathbf{B}$.

Применим процедуру `BasisColumnSpace` пакета `VectorSpaces` для решения задачи построения базиса в подпространстве $\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$. Данная процедура составляет фундаментальную матрицу F и матрицу Λ.

```
> restart; with(LinearAlgebra):
> read "D:VectorSpaces.m"; with(VectorSpaces):
> # Задаем исходную систему вектор-столбцов пространства
>a[1]:=Vector[column](5,[[-1],[2],[0],[-4],[3]]):
 …
 a[6]:=Vector[column](5,[[9],[-10],[20],[1],[8]]):
 S:={a[1], a[2], a[3], a[4], a[5], a[6]};
 A:=Matrix(n,k,[a[1], a[2], a[3], a[4], a[5], a[6]]):
```

$$S:=\left\{\begin{bmatrix}-1\\2\\0\\-4\\3\end{bmatrix},\begin{bmatrix}2\\-2\\6\\1\\-3\end{bmatrix},\begin{bmatrix}5\\-5\\13\\-1\\4\end{bmatrix},\begin{bmatrix}4\\-5\\7\\2\\4\end{bmatrix},\begin{bmatrix}3\\-4\\6\\5\\-6\end{bmatrix},\begin{bmatrix}9\\-10\\20\\1\\8\end{bmatrix}\right\}$$

```
> BasisColumnSpace(S);
```

$$F:=\begin{bmatrix}-1&2&5\\2&-2&-5\\0&6&13\\-4&1&-1\\3&-3&4\end{bmatrix}$$

$$\Lambda:=\begin{bmatrix}1&0&0&-1&-1&-1\\0&1&0&-1&1&-1\\0&0&1&1&0&2\end{bmatrix}$$

```
> # Второй способ: находим базис пространства при помощи процедуры ColumnSpace пакета LinearAlgebra
> Column_Basis:=ColumnSpace(A);
```

$$Column\_Basis:=\left[\begin{bmatrix}1\\0\\0\\\frac{-27}{2}\\\frac{69}{2}\end{bmatrix},\begin{bmatrix}0\\1\\0\\\frac{-35}{4}\\\frac{75}{4}\end{bmatrix},\begin{bmatrix}0\\0\\1\\\frac{7}{4}\\\frac{-23}{4}\end{bmatrix}\right]$$

```
> F1:=Matrix(n,r,[Column_Basis[1],Column_Basis[2],Column_Basis[3]]):
> # Составляем матрицу Lambda1 из координатных вектор-столбцов векторов исходной системы
> Lambda1:=LinearSolve(F1, A);
```

$$\Lambda 1 := \begin{bmatrix} -1 & 2 & 5 & 4 & 3 & 9 \\ 2 & -2 & -5 & -5 & -4 & -10 \\ 0 & 6 & 13 & 7 & 6 & 20 \end{bmatrix}$$

Рассмотрим решение задачи построения базиса конечномерного линейного пространства в СКА wxmaxima. В результате выполнения процедуры `columnspace(A)` с матрицей A выводится базис подпространства $W = \mathbf{Lin}\left(\overline{a_1}, \overline{a_2}, ..., \overline{a_6}\right)$ (результат выдается в виде линейной оболочки, порожденной базисными вектор-столбцами). С помощью фундаментальной матрицы F в результате решения системы линейных алгебраических уравнений определяется координатный вектор-столбец вектора $\overline{a_4}$ (переменная `y`).

```
(%i8)  kill(all); n:5$
       a[1]:[-1,2,0,-4,3]$ a[2]:[2,-2,6,1,-3]$
       a[3]:[5,-5,13,-1,4]$ a[4]:[4,-5,7,2,4]$
       a[5]:[3,-4,6,5,-6]$ a[6]:[9,-10,20,1,8]$
       A: transpose(matrix(a[1],a[2],a[3],a[4],a[5],a[6]));
(%o0)  done
```

$$(A) \quad \begin{bmatrix} -1 & 2 & 5 & 4 & 3 & 9 \\ 2 & -2 & -5 & -5 & -4 & -10 \\ 0 & 6 & 13 & 7 & 6 & 20 \\ -4 & 1 & -1 & 2 & 5 & 1 \\ 3 & -3 & 4 & 4 & -6 & 8 \end{bmatrix}$$

```
(%i9)  columnspace(A);
       /*нахождение базиса подпространства W, порожденного
       вектор-столбцами*/
0 errors, 0 warnings
```

$$(\%o9) \quad \mathrm{span}\left(\begin{bmatrix} -1 \\ 2 \\ 0 \\ -4 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ -2 \\ 6 \\ 1 \\ -3 \end{bmatrix}, \begin{bmatrix} 5 \\ -5 \\ 13 \\ -1 \\ 4 \end{bmatrix}\right)$$

```
(%i10) F: submatrix(A,4,5,6); /*фундаментальная матрица*/;
```

$$(F) \quad \begin{bmatrix} -1 & 2 & 5 \\ 2 & -2 & -5 \\ 0 & 6 & 13 \\ -4 & 1 & -1 \\ 3 & -3 & 4 \end{bmatrix}$$

```
(%i11)   for i:1 thru 5 do
           (var[i]:0,
           for j:1 thru 3 do
            var[i]:var[i]+F[i,j]*alpha[j],
           eq[i]:var[i]=a[4][i]);

(%o11)   done

(%i12)   [globalsolve: true, programmode: true];
(%o12)   [true,true]

(%i14)   sys:[eq[1], eq[2], eq[3], eq[4], eq[5]]$
         y: linsolve (sys, [alpha[1], alpha[2], alpha[3]]);
solve: dependent equations eliminated: (5 4)
(y)      [alpha1:-1,alpha2:-1,alpha3:1]
```

Переход от внутреннего способа описания подпространства к внешнему способу описания подпространства

Наиболее часто используют два способа описания подпространства в пространстве $\mathbf{R}^n$ вектор-столбцов. При первом способе (*внутренний способ*) подпространство описывается в виде линейной оболочки порождающих вектор-столбцов: $W=\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$ $\left(\overline{a_i}\in\mathbf{R}^n,\ \ i=\overline{1,k}\right)$. Второй способ (*внешний способ*) означает, что подпространство задается в виде однородной системы линейных алгебраических уравнений (ОСЛАУ): $W=\mathbf{L}_A^0=\left\{\overline{x}\in\mathbf{R}^n:\ \underset{m\times n}{A}\cdot\underset{n\times 1}{\overline{x}}=\overline{0}_m\right\}$. Способ задания подпространства зависит от конкретной решаемой задачи.

Рассмотрим задачу о переходе от внутреннего способа описания подпространства $W=\mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$ $\left(\overline{a_i}\in\mathbf{R}^n,\ \ i=\overline{1,k}\right)$ к внешнему способу. Требуется составить ОСЛАУ с матрицей A, множество решений которой совпадает с подпространством W.

Предположим, что $\dim W=r$ и $\mathbf{B}=\left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_r}\right)$ – базис в W (максимальная система линейно независимых порождающих вектор-столбцов). Составим $n\times r$ - фундаментальную матрицу $\underset{n\times r}{F}=\left(\overline{b_1}\ \ \overline{b_2}\ \ ...\ \ \overline{b_r}\right)$, столбцами которой являются вектор-столбцы $\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_r}$. Так как $\dim W=r$, то $\dim\mathbf{L}_A^0=r$. Следовательно, матрица A должна иметь размеры $\left(n-r\right)\times r$. При выполнении этого условия $rang A=n-r$ и фундаментальная система решений для ОСЛАУ $\underset{(n-r)\times n}{A}\cdot\underset{n\times 1}{\overline{x}}=\overline{0}_{n-r}$ будет иметь ровно $n-r$ линейно независимых решений. Тогда $\dim\mathbf{L}_A^0=n-\left(n-r\right)=r$.

Далее, если $\overline{x} \in W$ и $\overline{x} \in \mathbf{L}_A^0$, то обязательно должны выполняться условия

$$\underset{(n-r)\times n}{A} \cdot \underset{n\times 1}{\overline{b_i}} = \overline{0}_{n-r} \ (i = \overline{1,r}),$$

которые можно переписать в виде одного матричного уравнения

$$\underset{(n-r)\times n}{A} \cdot \underset{n\times r}{F} = \underset{(n-r)\times r}{O}.$$

Обозначим через $\overline{\alpha}^T = (\alpha_1 \quad \alpha_2 \quad \ldots \quad \alpha_n)$ вектор-строку искомой матрицы A. В силу последнего матричного уравнения каждая строка $\overline{\alpha}^T$ матрицы A должна удовлетворять и уравнению

$$\overline{\alpha}^T \cdot F = \overline{0}_r^T.$$

Перепишем его в виде

$$F^T \cdot \overline{\alpha} = \overline{0}_r.$$

Решив последнее уравнение относительно $\overline{\alpha}$, получим фундаментальную матрицу A размером $n \times (n-r)$. В результате искомая матрица A будет иметь вид $\underset{(n-r)\times n}{A} = \left(\underset{n\times(n-r)}{A} \right)^T$.

Процедура `InternalExternal` пакета `VectorSpaces` может быть применена для решения задачи о переходе от внутреннего способа задания подпространства к внешнему способу. В результате применения процедуры `InternalExternal` формируются базис **B** подпространства W и искомая матрица A ОСЛАУ.

```
> restart; with(LinearAlgebra): n:=5: k:=5:
> read "D:VectorSpaces.m"; with(VectorSpaces):
> a[1]:=Vector[column](n,[-1,1,0,1,2]):
 …
 a[5]:=Vector[column](n,[0,5,1,2,3]):
 S:=Matrix(n,k,[a[1],a[2],a[3],a[4],a[5]]);
```

$$S := \begin{bmatrix} -1 & 2 & 0 & 1 & 0 \\ 1 & 3 & -1 & 5 & 5 \\ 0 & 1 & -1 & 2 & 1 \\ 1 & 0 & 2 & -1 & 2 \\ 2 & -1 & 0 & 1 & 3 \end{bmatrix}$$

```
> # процедура перехода от способа задания подпространства в виде вектор-столбцов к описанию в виде ОСЛАУ
  InternalExternal(S);
```

$$Basis\_Lin := \left[\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ -\frac{5}{4} \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ \frac{3}{4} \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ -3 \\ -\frac{3}{4} \end{bmatrix} \right]$$

$$sys\_OSLAU := \left[\frac{5}{3} x_1 - x_2 + x_3 + \frac{4}{3} x_5 = 0, -5 x_1 + 2 x_2 + x_4 - 4 x_5 = 0 \right]$$

$$A := \begin{bmatrix} \frac{5}{3} & -1 & 1 & 0 & \frac{4}{3} \\ -5 & 2 & 0 & 1 & -4 \end{bmatrix}$$

Нахождение базиса в сумме и пересечении двух подпространств

Для построения базиса в сумме двух подпространств удобнее пользоваться внутренним способом описания подпространства. Пусть подпространства $W_1,\ W_2 \subset \mathbf{R}^n$ заданы в виде линейных оболочек порождающих вектор-столбцов $W_1 = \mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_{r1}}\right)$, $W_2 = \mathbf{Lin}\left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_{r2}}\right)$ $\left(\overline{a_i},\ \overline{b_j} \in \mathbf{R}^n,\ \ i = \overline{1,r1},\ j = \overline{1,r2}\right)$, причем $\mathbf{B}_1 = \left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_{r1}}\right)$, $\mathbf{B}_2 = \left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_{r2}}\right)$ есть базисы в $W_1,\ W_2$ соответственно $\left(r_i = \dim\left(W_i\right),\ i = \overline{1,2}\right)$.

Требуется построить базис $\mathbf{B}_3$ и найти размерность $r_3 = \dim\left(W_3\right)$ подпространства $W_3 = W_1 + W_2$.

Так как по определению сумма подпространств $W_1,\ W_2$ есть

$$W_3 = W_1 + W_2 = \left\{ \overline{z} \in V :\ \overline{z} = \overline{x} + \overline{y},\ \overline{x} \in W_1,\ \overline{y} \in W_2 \right\},$$

$$\overline{x} = \sum_{i=1}^{r1} \alpha_i \overline{a_i}\ \left(\alpha_i \in \mathbf{R},\ i = \overline{1,r1}\right),\ \overline{y} = \sum_{j=1}^{r2} \beta_j \overline{b_j}\ \left(\beta_j \in \mathbf{R},\ j = \overline{1,r2}\right),$$

то $\overline{z} = \overline{x} + \overline{y} = \sum_{i=1}^{r1} \alpha_i \overline{a_i} + \sum_{j=1}^{r2} \beta_j \overline{b_j}$. Значит, $W_3 = W_1 + W_2$ можно задать в виде линейной оболочки $W_3 = W_1 + W_2 = \mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_{r1}},\ \overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_{r2}}\right)$, порожденной векторами базисов $\mathbf{B}_1,\ \mathbf{B}_2$.

Для построения базиса в пересечении двух подпространств следует воспользоваться внешним способом описания подпространства. Пусть подпространства $W_1,\ W_2$ заданы в виде ОСЛАУ: $W_1 = \mathbf{L}^0_{A1},\ W_2 = \mathbf{L}^0_{A2}$.

Требуется построить базис $\mathbf{B}_0$ и найти размерность $r_0 = \dim\left(W_0\right)$ подпространства $W_0 = W_1 \cap W_2$. Так как по определению пересечения подпространств

$$\overline{x} \in W_0 = W_1 \cap W_2 \ \Leftrightarrow\ \begin{cases} \overline{x} \in W_1, \\ \overline{x} \in W_2, \end{cases} \ \Leftrightarrow\ \begin{cases} A_1 \cdot \overline{x} = \overline{0}, \\ A_2 \cdot \overline{x} = \overline{0}, \end{cases}$$

то вектор-столбец $\overline{x}$ должен являться решением объединенной ОСЛАУ

$$A \cdot \overline{x} = \overline{0}$$

с основной матрицей $A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = \left(A_1^T \middle| A_2^T \right)^T$.

Таким образом, подпространство $W_0 = W_1 \cap W_2$ можно задать в виде

$$W_0 = W_1 \cap W_2 = \mathbf{L}_A^0 = \left\{ \bar{x} : A \cdot \bar{x} = \bar{0} \right\}.$$

В следующем листинге процедура `SummIntersectBasis` пакета `VectorSpaces` применяется для построения базисов суммы, пересечения двух подпространств. Выводятся базисы подпространств W_1, W_2, $W_3 = W_1 + W_2$, $W_0 = W_1 \cap W_2$ и матрицы A_1, A_2 соответствующих однородных систем линейных алгебраических уравнений, определяющие внешний способ задания подпространств W_1, W_2.

```
> restart; with(LinearAlgebra): n:=5: k:=5:
> read "D:VectorSpaces.m"; with(VectorSpaces):
>a[1]:=Vector[column](n,[[1],[-1],[0],[2]]):
a[2]:=Vector[column](n,[[2],[3],[1],[-1]]):
a[3]:=Vector[column](n,[[5],[5],[2],[0]]):
S[1]:=Matrix(n,k[1],[a[1], a[2], a[3]]):
b[1]:=Vector[column](n,[[0],[1],[2],[-3]]):
b[2]:=Vector[column](n,[[-1],[-4],[-1],[3]]):
b[3]:=Vector[column](n,[[3],[-1],[2],[-2]]):
b[4]:=Vector[column](n,[[3],[0],[4],[-5]]):
S[2]:=Matrix(n,k[2],[b[1], b[2], b[3], b[4]]): {S[1],S[2]};
```

$$\left\{ \begin{bmatrix} 1 & 2 & 5 \\ -1 & 3 & 5 \\ 0 & 1 & 2 \\ 2 & -1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & -1 & 3 & 3 \\ 1 & -4 & -1 & 0 \\ 2 & -1 & 2 & 4 \\ -3 & 3 & -2 & -5 \end{bmatrix} \right\}$$

```
> SummIntersectBasis(S[1],S[2]);
```

$$Basis\_W_1 := \left[\begin{bmatrix} 1 \\ 0 \\ \frac{1}{5} \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ \frac{1}{5} \\ -1 \end{bmatrix} \right]$$

$$Basis\_W_2 := \left[\begin{bmatrix} 1 \\ 0 \\ 0 \\ \frac{1}{25} \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ \frac{-11}{25} \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ \frac{-32}{25} \end{bmatrix} \right]$$

$$Basis\_W_3 := \left[\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right]$$

$$A_1 := \begin{bmatrix} \frac{-1}{5} & \frac{-1}{5} & 1 & 0 \\ -1 & 1 & 0 & 1 \end{bmatrix}$$

$$A_2 := \begin{bmatrix} \frac{-1}{25} & \frac{11}{25} & \frac{32}{25} & 1 \end{bmatrix}$$

$$Basis\_W_0 := \begin{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 1 \\ -3 \end{bmatrix} \end{bmatrix}$$

Рассмотрим решение задачи о нахождении базиса суммы и пересечения двух подпространств в среде wxMaxima.

```
(%i11)  kill(all); n:4$
        a[1]:[1,-1,0,2]$ a[2]:[2,3,1,-1]$ a[3]:[5,5,2,0]$
        b[1]:[0,1,2,-3]$ b[2]:[-1,-4,-1,3]$ b[3]:[3,-1,2,-2]$
        b[4]:[3,0,4,-5]$ S1: transpose(matrix(a[1],a[2],a[3]))$
        S2: transpose(matrix(b[1],b[2],b[3],b[4]))$
        S:addcol(S1,S2);
(%o0)   done
```

$$(S) \quad \begin{bmatrix} 1 & 2 & 5 & 0 & -1 & 3 & 3 \\ -1 & 3 & 5 & 1 & -4 & -1 & 0 \\ 0 & 1 & 2 & 2 & -1 & 2 & 4 \\ 2 & -1 & 0 & -3 & 3 & -2 & -5 \end{bmatrix}$$

```
(%i12)  [columnspace(S1), columnspace(S2)];
        /*нахождение базиса подпространства W, порожденного
        вектор-столбцами*/
0 errors, 0 warnings
```

$$(\%o12) \quad [\,\mathrm{span}\left(\begin{bmatrix} 1 \\ -1 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 1 \\ -1 \end{bmatrix}\right), \mathrm{span}\left(\begin{bmatrix} -1 \\ -4 \\ -1 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \\ -3 \end{bmatrix}, \begin{bmatrix} 3 \\ -1 \\ 2 \\ -2 \end{bmatrix}\right)]$$

```
(%i14)  columnspace(S);
        /*базис суммы подпространств W1, W2*/
```

$$(\%o14) \quad \mathrm{span}\left(\begin{bmatrix} 0 \\ 1 \\ 2 \\ -3 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 3 \\ -1 \\ 2 \\ -2 \end{bmatrix}\right)$$

```
(%i17)  F1: matrix(a[1],a[2])$ /*фундаментальная матрица*/;
        /*переход от внутреннего способа описания подпространтсва
        к внешнему в W1*/
        for i:1 thru 2 do
          (var1[i]:0,
          for j:1 thru n do
           var1[i]:var1[i]+F1[i,j]*x[j],
          eq1[i]:var1[i]=0);
        sys1:[eq1[1],eq1[2]];
(%o16)  done
```

$$(sys1) \quad [\,2\,x_4 - x_2 + x_1 = 0\,,\; -x_4 + x_3 + 3\,x_2 + 2\,x_1 = 0\,]$$

```
(%i18)  y: linsolve (sys1,[x[1],x[2],x[3],x[4]]);
```

$$(y)\quad [x_1=-\frac{\%r2+5\,\%r1}{5},\ x_2=\frac{5\,\%r1-\%r2}{5},\ x_3=\%r2,\ x_4=\%r1]$$

```
(%i20)  FSR1[1]:[subst([%r1=1,%r2=0], rhs(y[1])), subst([%r1=1,%r2=0], rhs(y[2])),
        subst([%r1=1,%r2=0], rhs(y[3])), subst([%r1=1,%r2=0], rhs(y[4]))];
        FSR1[2]:[subst([%r1=0,%r2=1], rhs(y[1])), subst([%r1=0,%r2=1], rhs(y[2])),
        subst([%r1=0,%r2=1], rhs(y[3])), subst([%r1=0,%r2=1], rhs(y[4]))];
```

$$(\%o19)\quad [-1,1,0,1]$$

$$(\%o20)\quad [-\frac{1}{5},-\frac{1}{5},1,0]$$

```
(%i21)  A1:matrix(FSR1[1], FSR1[2]);
        /*матрица ОСЛАУ, соответствующая внешнему способу задания W1*/
```

$$(A1)\quad \begin{bmatrix} -1 & 1 & 0 & 1 \\ -\frac{1}{5} & -\frac{1}{5} & 1 & 0 \end{bmatrix}$$

```
(%i24)  F2: matrix(b[1],b[2],b[3])$ /*фундаментальная матрица*/;
        /*переход от внутреннего способа описания подпространтсва
        к внешнему в W1*/
        for i:1 thru 3 do
          (var2[i]:0,
          for j:1 thru n do
           var2[i]:var2[i]+F2[i,j]*u[j],
          eq2[i]:var2[i]=0);
        [eq2[1],eq2[2],eq2[3]];
```

$$(\%o23)\quad done$$

$$(\%o24)\quad [-3\,u_4+2\,u_3+u_2=0,\ 3\,u_4-u_3-4\,u_2-u_1=0,\ -2\,u_4+2\,u_3-u_2+3\,u_1=0]$$

```
(%i25)  [globalsolve: true, programmode: true];
```

$$(\%o25)\quad [true,true]$$

```
(%i27)  sys2:[eq2[1], eq2[2], eq2[3]]$
        v: linsolve (sys2, [u[1], u[2], u[3],u[4]]);
```

$$(v)\quad [u_1:-\frac{\%r3}{25},\ u_2:\frac{11\,\%r3}{25},\ u_3:\frac{32\,\%r3}{25},\ u_4:\%r3]$$

```
(%i28)  FSR2[1]:[subst([%r3=1], rhs(v[1])), subst([%r3=1], rhs(v[2])),
        subst([%r3=1], rhs(v[3])), subst([%r3=1], rhs(v[4]))];
```

$$(\%o28)\quad [-\frac{1}{25},\frac{11}{25},\frac{32}{25},1]$$

```
        A2:matrix(FSR2[1]);
        /*матрица ОСЛАУ, соответствующая внешнему способу задания W2*/
```

$$(A2)\quad \begin{bmatrix} -\frac{1}{25} & \frac{11}{25} & \frac{32}{25} & 1 \end{bmatrix}$$

```
(%i32)  A:transpose(addcol(transpose(A1),transpose(A2)))$
        for i:1 thru 3 do
          (var3[i]:0,
          for j:1 thru n do
           var3[i]:var3[i]+A[i,j]*z[j],
          eq3[i]:var3[i]=0);
        [eq3[1],eq3[2],eq3[3]];
(%o31)  done
(%o32)  [z4+z2-z1=0, z3-z2/5-z1/5=0, z4+32 z3/25+11 z2/25-z1/25=0]

(%i34)  sys3:[eq3[1], eq3[2], eq3[3]]$
        w: linsolve (sys3, [z[1], z[2], z[3],z[4]]);
(w)     [z1 : %r4, z2 : 4 %r4, z3 : %r4, z4 : -3 %r4]

(%i35)  Intersetion:[subst([%r4=1], rhs(w[1])), subst([%r4=1], rhs(w[2])),
        subst([%r4=1], rhs(w[3])), subst([%r4=1], rhs(w[4]))];
        /*нахождение базиса пересечения W1, W2*/
(Intersetion)  [1, 4, 1, -3]
```

При помощи процедуры `columnspace` определяются базисы каждого из подпространств W_1, W_2, $W_3 = W_1 + W_2$ (результат выдается в виде линейной оболочки порождающих вектор-столбцов). Для построения базиса пересечения $W_0 = W_1 \cap W_2$ двух подпространств происходит перейдем от внутреннего способа задания подпространств W_1, W_2 к внешнему способу, в результате чего формируются матрицы соответствующих однородных систем линейных алгебраических уравнений (матрицы `A1`, `A2`). Искомый базис пересечения двух подпространств $W_0 = W_1 \cap W_2$ определяется как решение системы $A \cdot \bar{x} = \bar{0}$ с основной матрицей $A = \left(A_1^T \middle| A_2^T \right)^T$ (переменная `Intersection`).

Нахождение инвариантных подпространств относительно линейного оператора

Пусть в конечномерном вещественном линейном пространстве V задан линейный оператор $\mathbf{A}: V \to V$. Тогда для каждого собственного значения $\lambda \in \mathbf{R}$ линейного оператора $\mathbf{A}: V \to V$ существует "расширяющая" цепочка инвариантных относительно этого оператора подпространств:

$$\{\theta\} \subset K_\lambda^{(1)} \subset K_\lambda^{(2)} \subset \ldots \subset K_\lambda^{(m)} \subset V,$$

где $K_\lambda^{(i)} = Ker\left\{ (\mathbf{A} - \lambda \mathbf{I})^i \right\}$ есть ядро линейного оператора $(\mathbf{A} - \lambda \mathbf{I})^i$ ($i = \overline{1, m}$, $m \le n$, $n = \dim V$). Инвариантное подпространство $K_\lambda^{(m)} = Ker\left\{ (\mathbf{A} - \lambda \mathbf{I})^m \right\}$ с наименьшим натуральным показателем m, для которого $K_\lambda^{(m)} = K_\lambda^{(m+1)}$, называется корневым подпространством.

Если все различные корни $\lambda_1, \lambda_2, ..., \lambda_k$ характеристического уравнения $\det(A-\lambda E)=0$ (A есть матрица линейного оператора $\mathbf{A}:V\to V$ в некотором базисе) являются собственными значениями оператора $\mathbf{A}:V\to V$, то пространство V можно разложить в прямую сумму корневых подпространств:

$$V=K_{\lambda_1}^{(m_1)}\oplus K_{\lambda_2}^{(m_2)}\oplus ...\oplus K_{\lambda_k}^{(m_k)}.$$

При этом существует базис линейного пространства V, в котором матрица A принимает блочно-диагональный вид $A=diag\{A_1, A_2, ..., A_k\}$, где A_i $\left(i=\overline{1,k}\right)$ есть матрица сужения $K_{\lambda_i}^{(m_i)}\to K_{\lambda_i}^{(m_i)}$ линейного оператора $\mathbf{A}:V\to V$ на корневое подпространство $K_{\lambda_i}^{(m_i)}$.

Процедура `Invariant_SubSpace` пакета `VectorSpaces` предназначена для нахождения инвариантных (корневых) подпространств относительно линейного оператора в вещественном линейном пространстве. Для каждого собственного значения линейного оператора процедура `Invariant_SubSpace` возвращает список в виде базисов инвариантных подпространств относительно этого оператора. Последний элемент списка есть базис корневого подпространства. Матрица `Block_Matrix` соответствует блочно-диагональной матрице $A=diag\{A_1, A_2\}$.

```
> restart; n:=5: with(LinearAlgebra): E:=IdentityMatrix(n):
  # формирование исходных данных (матрицы A)
  A:=Matrix(n,n,[[1,-3,1,2,4],[2,1,-1,-2,1],[-5,-1,-2,0,-4],
                 [2,0,2,0,0],[1,3,-1,-2,-2]]):
  Lambda:=Transpose(Eigenvalues(A));
```

$$\Lambda := [2, 2, -2, -2, -2]$$

```
> Invariant_SubSpace:=proc(A1::Matrix, lambda)
  local i,r,Dim,B,Basis_B,sys;
# процедура построения инвариантных (корневых) подпространств
for i from 1 by 1 to n do
 B:=(A1-lambda*E)^(i):
 r[i]:=Rank(B);
 Basis_B[i]:=NullSpace(B);
end do:
sys:=Basis_B[1];
for i from 2 by 1 while (r[i]<>r[i-1] and i<=n) do
 Dim[i]:=n-r[i];
 Basis_B[i]:=Basis_B[i];
 sys:=<sys|Basis_B[i]>;
end do;
RETURN(sys);
end proc:
> Inv_SubSpace1:=Invariant_SubSpace(A, -2);
```

$$Inv\_SubSpace1 := \left[\left\{ \begin{bmatrix} -1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \right\} \quad \left\{ \begin{bmatrix} -1 \\ 1 \\ \frac{4}{3} \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ \frac{-4}{3} \\ 1 \\ 0 \end{bmatrix} \right\} \quad \left\{ \begin{bmatrix} -1 \\ \frac{-1}{15} \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ \frac{16}{15} \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \frac{4}{5} \\ 1 \\ 0 \\ 0 \end{bmatrix} \right\} \right]$$

```
> Inv_SubSpace2:=Invariant_SubSpace(A, 2);
```

$$Inv\_SubSpace2 := \left[\left\{ \begin{bmatrix} -1 \\ 1 \\ 0 \\ -1 \\ 1 \end{bmatrix} \right\} \quad \left\{ \begin{bmatrix} \frac{64}{39} \\ \frac{16}{39} \\ \frac{-28}{13} \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \frac{25}{39} \\ \frac{55}{39} \\ \frac{-28}{13} \\ 0 \\ 1 \end{bmatrix} \right\} \right]$$

```
> #матрица T преобразования (столбцы матрицы - базисные векторы)
T:=Matrix(n,n,[Inv_SubSpace1[1][3][1],    Inv_SubSpace1[1][3][2],
Inv_SubSpace1[1][3][3],Inv_SubSpace2[1][2][1],
Inv_SubSpace2[1][2][2]]):
#блочно-диагональный вид матрицы линейного оператора в базисе
Block_Matrix:=MatrixInverse(T).A.T;
```

$$Block\_Matrix := \begin{bmatrix} \frac{-16}{5} & \frac{6}{5} & \frac{7}{5} & 0 & 0 \\ -2 & 0 & 2 & 0 & 0 \\ \frac{16}{15} & \frac{-16}{15} & \frac{-14}{5} & 0 & 0 \\ 0 & 0 & 0 & \frac{-40}{39} & \frac{-118}{39} \\ 0 & 0 & 0 & \frac{118}{39} & \frac{196}{39} \end{bmatrix}$$

2. Евклидовы пространства. Метод наименьших квадратов

Основной круг задач по теме “Евклидовы пространства” касается задания скалярного произведения различными способами, нахождения матрицы Грама системы векторов, построения ортонормированного базиса (проведения процесса ортогонализации Грама - Шмидта), нахождения ортогонального дополнения к подпространству, задачи о перпендикуляре. В качестве приложения рассматривается решение многомерной задачи о наименьших квадратах.

Пакет `EuclideanSpace` СКА Maple содержит ряд процедур для решения перечисленных выше задач (см. табл. 2).

```
> read "D:EuclideanSpace.m"; with(EuclideanSpace);
```

[*GramSchmidtMatrix*, *GramSchmidtPolynom*, *GramSchmidtStandard*, *LeastSquares2*, *MatrixGram*, *OrthogonalComplement*, *Perpendicular* , *ScalarProductBilForm* , *ScalarProductPolynom* , *VerificationScalarProduct*]

Таблица 2

GramSchmidtMatrix	Процедура построения ортонормированного базиса системы арифметических векторов, если скалярное произведение в пространстве векторов задается в виде симметрической билинейной формы (симметрической положительно определенной квадратной матрицы S): $(\bar{x}, \bar{y}) = \bar{x}^T \cdot \underset{n\times n}{S} \cdot \bar{y} = \sum_{i=1}^{n}\sum_{j=1}^{n} s_{ij} x_i y_j$
GramSchmidtPolynom	Процедура построения ортонормированного базиса системы многочленов в пространстве $\mathbf{P}_n[t]$. Скалярное произведение многочленов $x_n(t) = \sum_{i=0}^{n} a_i t^i$, $y_n(t) = \sum_{i=0}^{n} b_i t^i$ задается в виде $(x_n(t), y_n(t)) = \sum_{i=0}^{n} a_i b_i$
GramSchmidtStandard	Процедура построения ортонормированного базиса системы арифметических векторов в случае стандартного задания скалярного произведения $(\bar{x}, \bar{y}) = \bar{x}^T \cdot \bar{y} = \sum_{i=1}^{n} x_i y_i$
MatrixGram	Построение матрицы Грама системы векторов
LeastSquares2	Решение задачи о наименьших квадратах
OrthogonalComplement	Процедура построения ортогонального дополнения к подпространству
Perpendicular	Решение задачи о перпендикуляре
ScalarProductBilForm	Процедура нахождения скалярного произведения арифметических векторов в виде симметрической билинейной формы (симметрической положительно определенной квадратной матрицы)

ScalarProductPolynom	Процедура нахождения скалярного произведения многочленов
VerificationScalarProduct	Логическая функция. Проверяет, определяет ли заданная матрица скалярное произведение в виде билинейной формы (проверяет аксиомы скалярного произведения)

Процесс ортогонализации Грама - Шмидта

В любом конечномерном евклидовом пространстве существует ортонормированный базис. Данный базис можно построить, проведя процесс ортогонализации Грама - Шмидта по заданному произвольному базису евклидова пространства: если задан базис $\mathbf{B}=(a_1,\ a_2,\ ...,\ a_n)$ евклидова пространства, то процесс ортогонализации – получения ортонормированного базиса $\mathbf{B}^{(0)}=(e_1,\ e_2,\ ...,\ e_n)$ заключается в последовательном вычислении следующих векторов [(a_i, a_j) есть скалярное произведение векторов a_i, a_j, а $\|a\|$ – евклидова норма вектора a]:

$$e_1=\frac{g_1}{\|g_1\|},\ g_1=a_1,\ \|e_1\|=1,$$

$$e_2=\frac{g_2}{\|g_2\|},\ g_2=a_2-(a_2,e_1)\cdot e_1,,\ \|e_2\|=1,$$

$$e_3=\frac{g_3}{\|g_3\|},\ g_3=a_3-(a_3,e_1)\cdot e_1-(a_3,e_2)\cdot e_2,\ \|e_3\|=1,$$

.....

$$e_k=\frac{g_k}{\|g_k\|},\ g_k=a_k-(a_k,e_1)\cdot e_1-...-(a_k,e_{k-1})\cdot e_{k-1},,\ \|e_k\|=1,\ 4\le k\le n,$$

где $(e_i, e_j)=0$ при всех $i\neq j,\ i,j\in\{1,...,n\}$.

Пусть требуется провести процесс ортогонализации Грама - Шмидта системы векторов базиса $\mathbf{B}$ в пространстве $\mathbf{R}^n$, составить соответствующую ортогональную матрицу U, если скалярное произведение задается стандартным образом: $(\bar{x},\ \bar{y})=\bar{x}^T\cdot\bar{y}=\sum_{i=1}^{n}x_iy_i$ и в виде симметрической билинейной формы $(x,\ y)=x^T\cdot S\cdot y$. Для решения задачи можно применить процедуры GramSchmidtMatrix, GramSchmidtStandard, ScalarProductBilForm, VerificationScalarProduct пакета EuclideanSpace.

```
> restart; with(LinearAlgebra):
> read "D:EuclideanSpace.m"; with(EuclideanSpace):
```

```
> n:=3: #задаем размерность евклидового пространства
> a[1]:=Vector[column](n, [1,0,1]):
  a[2]:=Vector[column](n, [-1,1,0]):
  a[3]:=Vector[column](n, [0,2,1]): A:=a[1]:
> for i from 2 by 1 to n do
    A:=<A|a[i]>; #собираем вектор-столбцы в матрицу
  end do:
> Ortonorm_Basis:=GramSchmidtStandard(A);
#вывод искомого ортонормированного базиса пространства
```

$$Ortonorm\_Basis := \left[\begin{bmatrix} \frac{\sqrt{2}}{2} \\ 0 \\ \frac{\sqrt{2}}{2} \end{bmatrix}, \begin{bmatrix} -\frac{\sqrt{6}}{6} \\ \frac{\sqrt{6}}{3} \\ \frac{\sqrt{6}}{6} \end{bmatrix}, \begin{bmatrix} \frac{\sqrt{3}}{3} \\ \frac{\sqrt{3}}{3} \\ -\frac{\sqrt{3}}{3} \end{bmatrix}\right]$$

```
> # Собираем ортонормированные векторы в ортогональную матрицу
> Ortogonal_Matrix:=Ortonorm_Basis[1]:
> for i from 2 by 1 to n do
    Ortogonal_Matrix:=<Ortogonal_Matrix|Ortonorm_Basis[i]>;
  end do:
> U:=Ortogonal_Matrix; #выводим ортогональную матрицу U
```

$$U := \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{6}}{6} & \frac{\sqrt{3}}{3} \\ 0 & \frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{6}}{6} & -\frac{\sqrt{3}}{3} \end{bmatrix}$$

```
> bool:=IsOrthogonal(U); #проверяем матрицу U на ортогональность
```

$$bool := true$$

```
> Процесс ортогонализации Грама - Шмидта, если скалярное произведение задается в виде симметрической билинейной формы
> S:=Matrix(n,n,[[1,1,0], [1,2,0], [0,0,1]]):
> bool:=VerificationScalarProduct(S);
#проверяем, задает ли матрица S скалярное произведение
```

$$bool := true$$

```
> Ortonorm_Basis2:=GramSchmidtMatrix(A,S);
# выводим искомый ортонормированный базис относительно симметрической билинейной формы
```

$$Ortonorm\_Basis2 := \left[\begin{bmatrix} \frac{\sqrt{2}}{2} \\ 0 \\ \frac{\sqrt{2}}{2} \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \frac{\sqrt{2}}{2} \\ 0 \\ -\frac{\sqrt{2}}{2} \end{bmatrix}\right]$$

```
> Ortogonal_Matrix2:=Ortonorm_Basis2[1]:
> for i from 2 by 1 to n do
    Ortogonal_Matrix2:=<Ortogonal_Matrix2|Ortonorm_Basis2[i]>;
 end do:
> U2:=Ortogonal_Matrix2: Transpose(U2).S.U2;
```

#проверяем матрицу U2 на ортогональность относительно симметрической билинейной формы

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Ортогональное дополнение к подпространству.
Задача о перпендикуляре

Напомним основные понятия, которые понадобятся в данном пункте.

Ортогональным дополнением $W^{\perp}$ линейного подпространства W в евклидовом пространстве V называется множество векторов $w^{\perp} \in V$, ортогональных каждому вектору $w \in W$: $W^{\perp} = \left\{ w^{\perp} \in V : \ \left(w^{\perp}, \ w \right) = 0, \ w \in W \right\}$.

Ортогональное дополнение $W^{\perp}$ линейного подпространства W в евклидовом пространстве V является подпространством, причем

$$W \oplus W^{\perp} = V, \ \dim W + \dim W^{\perp} = \dim V .$$

Каково бы ни было подпространство W в евклидовом пространстве V, любой вектор $x \in V$ можно однозначно представить в виде ортогонального разложения на две составляющие:

$$x = w + w^{\perp}, \ w \in W, \ w^{\perp} \in W^{\perp},$$

где $w \in W$, $w^{\perp} \in W^{\perp}$ называются соответственно ортогональной проекцией и ортогональной составляющей вектора $x \in V$ на подпространство W.

Пусть W в евклидовом пространстве V имеет базис $\mathbf{B} = \left\{ f_1, \ f_2, \ ...., \ f_r \right\}$ $(r = \dim W)$. Так как ортогональное дополнение $W^{\perp}$ является подпространством и имеет размерность $\dim W^{\perp} = n - r$ $(n = \dim V)$, то оно также обладает базисом $\mathbf{B}^{\perp} = \left\{ g_1, \ g_2, \ ...., \ g_{n-r} \right\}$, причем $\left(f_i, \ g_j \right) = 0$ при всех $i = \overline{1, r}, \ j = \overline{1, n-r}$.

Рассмотрим два основных способа построения ортогонального дополнения к подпространству, заданному двумя способами (внешним и внутренним).

Пусть задана матрица A размерОМ $n \times k$:

$$\underset{n \times k}{A} = \left(\overline{a_1} \middle| \overline{a_2} \middle| ... \middle| \overline{a_k} \right) = \begin{pmatrix} a_1^{(1)} & a_1^{(2)} & ... & a_1^{(k)} \\ a_2^{(1)} & a_2^{(2)} & ... & a_2^{(k)} \\ ... & ... & ... & ... \\ a_n^{(1)} & a_n^{(2)} & ... & a_n^{(k)} \end{pmatrix},$$

где $\overline{a_i} = \left(a_1^{(i)} \quad a_2^{(i)} \quad ... \quad a_n^{(i)} \right)^T \in \mathbf{R}^n$ $(i = \overline{1, k})$.

Если подпространство W описывается внутренним способом $W = \mathbf{Lin}\left(\overline{a_1},\ \overline{a_2},\ ...,\ \overline{a_k}\right)$ в виде линейной оболочки порождающих вектор-столбцов, то $W^{\perp} = L^0_{A^T} = \left\{\overline{x} \in \mathbf{R}^n :\ \underset{k\times n}{A^T} \cdot \overline{x} = \overline{0}_k\right\}$ решений системы $\underset{k\times n}{A^T} \cdot \overline{x} = \overline{0}_k$ является ортогональным дополнением к W.

Если W задано внешним способом $W = L^0_A = \left\{\overline{x} \in \mathbf{R}^n :\ \underset{k\times n}{A} \cdot \overline{x} = \overline{0}_k\right\}$, то ортогональное дополнение $W^{\perp}$ к W описывается в виде линейной оболочки порождающих вектор-столбцов матрицы $\underset{n\times k}{A^T}$: $W^{\perp} = \mathbf{Lin}\left(\overline{b_1},\ \overline{b_2},\ ...,\ \overline{b_k}\right)$, где $\overline{b_i}$ – i-й столбец матрицы $\underset{n\times k}{A^T}$ $(i = \overline{1,k})$.

Задача о перпендикуляре ставится следующим образом. В евклидовом пространстве V заданы подпространство W и вектор $x \in V$. Требуется найти ортогональную проекцию $w \in W$ вектора x на подпространство W и его ортогональную составляющую $w^{\perp} \in W^{\perp}$. Алгоритм решения задачи следующий.

1. Находим произвольный базис $\mathbf{B} = \left(f_1,\ f_2,\ ...,\ f_r\right)$ в W $(r = \dim W)$. При помощи процесса ортогонализации Грама - Шмидта строим ортонормированный базис $\mathbf{B}_0 = \left(e_1,\ e_2,\ ....,\ e_r\right)$.

2. Находим разложение вектора $w \in W$ по базису $\mathbf{B}_0 = \left(e_1,\ e_2,\ ....,\ e_r\right)$. Оно имеет вид $w = w_1e_1 + w_2e_2 + ... + w_re_r = \sum_{i=1}^{r} w_ie_i$. При этом ортогональная составляющая $w^{\perp} \in W^{\perp}$ имеет вид

$$w^{\perp} = x - w = x - \sum_{i=1}^{r} w_ie_i\,.$$

Умножая последнее равенство скалярно на вектор e_j $\left(j = \overline{1,\ r}\right)$ и учитывая при этом, что $\left(w^{\perp},\ e_j\right) = 0$ $\left(j = \overline{1,\ r}\right)$, получАЕм

$$\left(x,\ e_j\right) - \sum_{i=1}^{r} w_i\left(e_i,\ e_j\right) = 0 \ \Leftrightarrow\ \left(x,\ e_j\right) - w_j = 0 \ \Leftrightarrow\ w_j = \left(x,\ e_j\right),\ \left(j = \overline{1,\ r}\right).$$

Итак, ортогональная проекция $w \in W$ имеет вид

$$w = \sum_{i=1}^{r} w_ie_i = \sum_{i=1}^{r} \left(x,\ e_i\right)e_i.$$

3. Ортогональную составляющую – вектор $w^{\perp} \in W^{\perp}$ вычисляем в виде

$$w^{\perp} = x - w = x - \sum_{i=1}^{r} w_ie_i\,.$$

Рассмотрим реализацию задачи о перпендикуляре в СКА Maple. Подпространство $W \subset \mathbf{R}^n$ задано внешним образом (матрицей однородной системЫ линейных алгебраических уравнений). Требуется найти

ортогональную проекцию $\bar{w} \in W$ вектора $\bar{x}$ на подпространство W и его ортогональную составляющую $\bar{w}^{\perp} \in W^{\perp}$.

```
> restart; with(LinearAlgebra):
> read "D:EuclideanSpace.m"; with(EuclideanSpace):
>  n:=4: #n-размерность пространства
>              A:=Matrix(m,n,[[2,1,2,-5],[3,2,2,-9],[1,0,2,-1]]):
  x:=Vector[column](n, [1,2,0,-1]):
> (Ortogonal_Pr,Ortogonal_Sost):=Perpendicular(A,x);
 #используется процедура для решения задачи о перпендикуляре
 #формируется ортонормированный базис B0 подпространства W
 #выводятся ортогональная проекция и ортогональная составляющая
```

$$B0 := \left[\begin{bmatrix} -\frac{2}{3} \\ \frac{2}{3} \\ \frac{1}{3} \\ 0 \end{bmatrix}, \begin{bmatrix} \frac{17\sqrt{83}}{249} \\ \frac{19\sqrt{83}}{249} \\ -\frac{4\sqrt{83}}{249} \\ \frac{3\sqrt{83}}{83} \end{bmatrix} \right]$$

$$Ortogonal\_Pr := \begin{bmatrix} \frac{50}{83} \\ \frac{134}{83} \\ -\frac{2}{83} \\ \frac{46}{83} \end{bmatrix} \qquad Ortogonal\_Sost := \begin{bmatrix} \frac{33}{83} \\ \frac{32}{83} \\ \frac{2}{83} \\ -\frac{129}{83} \end{bmatrix}$$

```
> DotProduct(Ortogonal_Pr, Ortogonal_Sost);
#проверяем условие ортогональности векторов
```

$$0$$

Многомерные задачи о наименьших квадратах

Рассмотрим решение многомерной задачи о наименьших квадратах. Пусть задана несовместная СЛАУ вида

$$A\bar{x} = \bar{b}, \tag{5.5}$$

где $A_{m \times n} = (a_{ij})$ $(i = \overline{1,\ m},\ j = \overline{1,\ n})$, $\bar{b} = (b_1 \quad \dots \quad b_m)^T$, $\bar{x} = (x_1 \quad \dots \quad x_n)^T$.

Несовместность системы (5.5) означает, что не существует такого вектор-столбца $\bar{x}$, что вектор-столбец $\bar{b}$ является линейной комбинацией столбцов матрицы A. Для каждого набора значений $x_1, \dots, x_n$ составим вектор невязки $\bar{d} = A\bar{x} - \bar{b}$ для системы уравнений (5.5). Функция

$$f(\bar{x}) = \sum_{i=1}^{m} (b_i - a_{i1}x_1 - \dots - a_{in}x_n)^2 \tag{5.6}$$

строго положительна, так как система (5.5) несовместна. Эту функцию можно рассматривать как оценку отклонения набора значений неизвестных $x_1, \dots, x_n$ от решения системы (5.5). Поскольку (5.5) несовместна, то возникает

задача нахождения вектора $\bar{x}^{(0)} \in \mathbf{R}^n$ оптимального приближения решения для системы (5.5), минимизирующего функцию (5.6).

Минимизация ошибки проводится по методу наименьших квадратов (МНК). Дадим алгебро-геометрическую интерпретацию поставленной задачи. Введем в рассмотрение вектор-столбцы основной матрицы системы (5.5):

$$\bar{a}_i = \begin{pmatrix} a_{1i} & \dots & a_{mi} \end{pmatrix}, \quad \underset{m\times n}{A} = \left(\bar{a}_1 \middle| \bar{a}_2 \middle| \dots \middle| \bar{a}_n\right).$$

С геометрической точки зрения задача нахождения вектора $\bar{x}^{(0)} \in \mathbf{R}^n$ по МНК, при котором функция (5.6) принимает наименьшее значение, равносильна задаче нахождениЯ вектора $\bar{p} = A\bar{x} \in \mathbf{Lin}\left(\bar{a}_1, \dots, \bar{a}_n\right)$, ближайшего к $\bar{b}$ по сравнению с остальными векторами из подпространства $W \equiv \mathbf{Lin}\left(\bar{a}_1, \dots, \bar{a}_n\right)$. При этом вектор $\bar{d}$ должен быть ортогонален к подпространству W. В результате получаем систему нормальных уравнений

$$A^T A\bar{x} = A^T \bar{b}. \qquad (5.7)$$

При этом система нормальных уравнений (5.7) всегда совместна.

Как известно, если система вектор-столбцов $\bar{a}_1, \dots, \bar{a}_n$ линейно независима, то единственный вектор оптимального приближения для системы (5.5) удовлетворяет равенству $\bar{x}^{(0)} = \left(A^T A\right)^{-1} A^T b$.

Пусть система вектор-столбцов $\bar{a}_1, \dots, \bar{a}_n$ является линейно зависимой. Вектор, являющийся решением системы нормальных уравнений (5.7) и имеющий минимальную норму из всех векторов, являющихся решениями системы (5.7), называется нормальным псевдорешением системы (5.5).

Пространство $\mathbf{R}^n$ представим в виде прямой суммы

$$\mathbf{R}^n = \mathbf{L}_A^0 \oplus \mathbf{Lin}\left(\bar{b}_1, \dots, \bar{b}_m\right),$$

где $\mathbf{L}_A^0 = \left\{x \in \mathbf{R}^n : A\bar{x} = \bar{0}_m\right\}$ – нуль-пространство матрицы A, $\mathbf{Lin}\left(\bar{b}_1, \dots, \bar{b}_m\right)$ – подпространство, порожденное вектор-столбцами матрицы A^T, $\bar{b}_i - i$-й столбец матрицы A^T $(i = \overline{1,m})$, причем $\mathbf{Lin}\left(\bar{b}_1, \dots, \bar{b}_m\right)$ является ортогональным дополнением к $\mathbf{L}_A^0$.

Представим решение $\bar{x}$ системы нормальных уравнений (5.7) в виде суммы:

$$\bar{x} = \bar{x}_{row} + \bar{x}_N, \qquad (5.8)$$

где $\bar{x}_N \in \mathbf{L}_A^0$, $\bar{x}_{row} \in \mathbf{Lin}\left(\bar{b}_1, \dots, \bar{b}_m\right)$. При этом:

1) вектор-столбец $\bar{x}_{row}$ есть решение системы $A\bar{x} = \bar{p}$,

2) так как общее решение системы $A\bar{x} = \bar{p}$ есть сумма общего решения системы $A\bar{x} = \bar{0}_m$ и частного решения системы $A\bar{x} = \bar{p}$, то все решения системы

(5.3), имеющие вид (5.8), имеют одни и те же компоненты из пространства $\mathbf{Lin}\left(\overline{b_1},\ ...,\ \overline{b_m}\right)$ и отличаются только компонентами из нуль-пространства $\mathbf{L}_A^0$.

Любая несовместная система уравнений (5.5) имеет единственное нормальное псевдорешение.

Общее решение $\overline{x} = X\left(c_1,\ ...,\ c_{n-r}\right)$ системы (5.7) представим в виде

$\overline{x} = X\left(c_1,\ ...,\ c_{n-r}\right) = \overline{y} + c_1\overline{E}_1 + ... + c_{n-r}\overline{E}_{n-r}$,

где $c_1,\ ...,\ c_{n-r} \in \mathbf{R}$, $r = rang\left(A^T A\right) = rangA$, $\dim \mathbf{L}_A^0 = n - r$, $\overline{y} = X\left(0,\ ...,\ 0\right)$, вектор-столбцы $\overline{E_1},\ ...,\ \overline{E_{n-r}} \in \mathbf{L}_A^0$ представляют собой фундаментальную систему решений однородной системы $\left(A^T A\right)\overline{x} = \overline{0}_n$.

Пусть $\mathbf{B}_0 = \left(\overline{g_1},\ ...,\ \overline{g_r}\right)$ есть базис подпространства $\mathbf{Lin}\left(\overline{b_1},\ ...,\ \overline{b_m}\right)$. Так как $\overline{x} \in \mathbf{Lin}\left(\overline{b_1},\ ...,\ \overline{b_m}\right)$, то $\overline{x} = \alpha_1\overline{g_1} + ... + \alpha_r\overline{g_r}$, где $\alpha_1,\ ...,\ \alpha_r$ – коэффициенты разложения вектора $\overline{x}$ по векторам базиса $\mathbf{B}_0$.

В результате получаем систему

$$\begin{cases} \overline{x} = \alpha_1\overline{g_1} + ... + \alpha_r\overline{g_r}, \\ \overline{x} = \overline{y} + c_1\overline{E}_1 + ... + c_{n-r}\overline{E}_{n-r} \end{cases}$$

относительно n неизвестных $c_1,\ ...,\ c_{n-r},\ \alpha_1,\ ...,\ \alpha_r$, которую можно переписать в виде

$$\alpha_1\overline{g_1} + ... + \alpha_r\overline{g_r} - c_1\overline{E}_1 - ... - c_{n-r}\overline{E}_{n-r} = \overline{y}. \qquad (5.9)$$

Можно доказать, что система (5.9) имеет единственное решение, то есть числа $\alpha_1,\ ...,\ \alpha_r$ определяют коэффициенты разложения вектора $\overline{x}$ по $\mathbf{B}_0$.

Процедура `LeastSquares2 из пакета EuclideanSpace` СКА Maple позволяет решить задачу нахождения нормального псевдорешения в случае линейной зависимости вектор-столбцов $\overline{a_1},\ ...,\ \overline{a_n}$ основной матрицы системы (5.5). В процессе использования процедуры на экран выводятся промежуточные результаты.

```
> restart; with(LinearAlgebra):
> read "D:EuclideanSpace.m"; with(EuclideanSpace):
>m:=4: n:=4:
>a[1]:=Vector[column](m,[1,0,2,1]):
 a[2]:=Vector[column](m,[1,2,4,-1]):
 a[3]:=Vector[column](m,[-1,2,0,-3]):
 a[4]:=Vector[column](m,[2,2,6,0]):
 A:=Matrix(m,n,[a[1],a[2],a[3],a[4]]);
 b:=Vector[column](m,[-1,2,0,-1]);
>LinearSolve(A,b);
 #проверка исходной линейной системы на совместность
   Error, (in LinearAlgebra:-LA_Main:-LinearSolve) inconsistent system
>LeastSquares2(A,b);
# общее решение X системы нормальных уравнений
```

```
# вектор-столбец y, фундаментальная система решений
# базис B0 подпространства строк матрицы A
# решение системы относительно (вектор alpha)
# вывод вектора x оптимального приближения
```

$$X := \begin{bmatrix} -\frac{29}{17} + c_2 + 3\,c_3 \\ c_2 \\ c_3 \\ \frac{10}{17} - c_2 - c_3 \end{bmatrix} \quad y := \begin{bmatrix} \frac{-29}{17} \\ 0 \\ 0 \\ \frac{10}{17} \end{bmatrix}$$

$$Fundamental\_Solve := \begin{bmatrix} 1 \\ 1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 3 \\ 0 \\ 1 \\ -1 \end{bmatrix}$$

$$Basis\_A := \left[\begin{bmatrix} 1 \\ 0 \\ -2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right]$$

$$\alpha := \left[\frac{-47}{289}, \frac{41}{289}, \frac{41}{289}, \frac{135}{289} \right]$$

$$X\_optimal := \left[\frac{-47}{289}, \frac{41}{289}, \frac{135}{289}, \frac{-6}{289} \right]$$

```
> Transpose(LeastSquares(A,b,optimize));
#проверка при помощи процедуры LeastSquares пакета LinearAlgebra
для определения вектора оптимального приближения по МНК
```

$$\left[\frac{-47}{289}, \frac{41}{289}, \frac{135}{289}, \frac{-6}{289} \right]$$

Рассмотрим реализацию задачи о нахождении нормального псевдорешения системы (5.5) в случае линейной зависимости вектор-столбцов основной матрицы системы в СКА wxMaxima.

```
kill(all); n:4$
A: matrix([1,1,-1, 2], [0,2,2,2], [2,4,0,6],[1,-1,-3,0])$
B: matrix([-1], [2],[0],[-1])$
```

(%o0) done

```
x:invert(transpose(A).A).transpose(A).B;
/*проверка САЛУ на совместность*/
```

```
expt: undefined: 0 to a negative exponent.
 -- an error. To debug this try: debugmode(true);
```

```
C:(transpose(A).A)$ D: transpose(A).B$
/*задание основной матрицы C и вектор-столбца D
правой части системы нормальных уравнений*/
```

```
/*построение системы нормальных уравнений (3)*/
for i:1 thru n do
  (var[i]:0,
  for j:1 thru n do
   var[i]:var[i]+C[i,j]*x[j],
  eq[i]:var[i]=D[i,1])$
```

```
[globalsolve: true, programmode: true];
```

(%o8) [true, true]

```
sys:[eq[1], eq[2], eq[3], eq[4]]; y: linsolve (sys, [x[1], x[2], x[3], x[4]]);
/*нахождение общего решения системы нормальных уравнений (3)*/
```

solve: dependent equations eliminated: (3 4)

(y) $$[x_1 : -\frac{-34\ \%r2+17\ \%r1+19}{17}, x_2 : -\frac{17\ \%r2+17\ \%r1-10}{17}, x_3 : \%r2, x_4 : \%r1]$$

```
(%i11) Y_Chastnoe_NSLAU:[subst([%r1=0,%r2=0], rhs(y[1])),
subst([%r1=0,%r2=0], rhs(y[2])),
subst([%r1=0,%r2=0], rhs(y[3])),
subst([%r1=0,%r2=0], rhs(y[4]))];
```

(Y_Chastnoe_NSLAU) $$[-\frac{19}{17}, \frac{10}{17}, 0, 0]$$

```
(%i13) /*нахождение общего решения однородной системы,
 соответствующей системе нормальных уравнений*/
NullVector:matrix([0],[0],[0],[0])$
for i:1 thru n do
  (var[i]:0,
  for j:1 thru n do
   var[i]:var[i]+C[i,j]*z[j],
  eqo[i]:var[i]=NullVector[i,1])$
```

```
(%i14) u: linsolve([eqo[1],eqo[2],eqo[3],eqo[4]],[z[1],z[2],z[3],z[4]]);
```

solve: dependent equations eliminated: (3 4)

(u) $$[z_1 : 2\ \%r4-\%r3, z_2 : -\%r4-\%r3, z_3 : \%r4, z_4 : \%r3]$$

```
(%i17)  /*нахождение ФСР, построение матрицы U*/
        FSR_OSLAU[1]:[subst([%r3=1,%r4=0], rhs(u[1])),
        subst([%r3=1,%r4=0], rhs(u[2])),
        subst([%r3=1,%r4=0], rhs(u[3])),
        subst([%r3=1,%r4=0], rhs(u[4]))]$
        FSR_OSLAU[2]:[subst([%r3=0,%r4=1], rhs(u[1])),
        subst([%r3=0,%r4=1], rhs(u[2])),
        subst([%r3=0,%r4=1], rhs(u[3])),
        subst([%r3=0,%r4=1], rhs(u[4]))]$
        U:transpose(matrix(FSR_OSLAU[1], FSR_OSLAU[2]));
```

(U)
$$\begin{bmatrix} -1 & 2 \\ -1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

```
(%i18)  /*нахождение базиса подпространства строк матрицы A*/
        Basis_A: columnspace(transpose(A));
```

```
0 errors, 0 warnings
```

(Basis_A)
$$\text{span}\left(\begin{bmatrix} 0 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ -1 \\ 2 \end{bmatrix}\right)$$

```
(%i21)  B0[1]: [0,2,2,2]$ B0[2]:[1,1,-1,2]$
        Matrix_B0: transpose(matrix(B0[1],B0[2]))$
```

```
(%i22)  F: matrix([Matrix_B0[1,1],Matrix_B0[1,2],-U[1,1],-U[1,2]],
                  [Matrix_B0[2,1],Matrix_B0[2,2],-U[2,1],-U[2,2]],
                  [Matrix_B0[3,1],Matrix_B0[3,2],-U[3,1],-U[3,2]],
                  [Matrix_B0[4,1],Matrix_B0[4,2],-U[4,1],-U[4,2]]);
```

(F)
$$\begin{bmatrix} 0 & 1 & 1 & -2 \\ 2 & 1 & 1 & 1 \\ 2 & -1 & 0 & -1 \\ 2 & 2 & -1 & 0 \end{bmatrix}$$

```
(%i24)  /*нахождение решения системы (5)*/
        for i:1 thru n do
          (var[i]:0,
          for j:1 thru n do
           var[i]:var[i]+F[i,j]*v[j],
          eq[i]:var[i]=Y_Chastnoe_NSLAU[i])$
        p:linsolve([eq[1],eq[2],eq[3],eq[4]],[v[1],v[2],v[3],v[4]]);
```

(p)
$$[v_1 : \frac{44}{289}, v_2 : -\frac{47}{289}, v_3 : -\frac{6}{289}, v_4 : \frac{135}{289}]$$

```
(%i25)  /*построение вектора оптимального приближения*/
        X_optimal: rhs(p[1]).B0[1]+rhs(p[2]).B0[2];
```

$$(X\_optimal)\quad \left[-\frac{47}{289},\frac{41}{289},\frac{135}{289},-\frac{6}{289}\right]$$

В программе командой `x:invert(transpose(A).A).transpose(A).b;` предпринимается попытка решения системы нормальных уравнений (5.7) матричным способом, вследствие чего выдается сообщение об ошибке:

```
expt: undefined: 0 to a negative exponent.
 -- an error. To debug this try: debugmode(true);
```

Далее находится общее решение системы нормальных уравнений (5.7), записанной в координатной форме (переменная `y`):

$$(y)\quad \left[x_1:-\frac{-34\,\%r2+17\,\%r1+19}{17},\ x_2:-\frac{17\,\%r2+17\,\%r1-10}{17},\ x_3:\%r2,\ x_4:\%r1\right]$$

и соответствующее частное решение (переменная `Y_Chastnoe_NSLAU`).

При помощи команды `U:transpose(matrix(FSR_OSLAU[1], FSR_OSLAU[2]));` формируется фундаментальная матрица U однородной системы, соответствующей системе нормальных уравнений (5.7). Применение команды `Basis_A: column(transpose(A));` дает базис подпространства строк матрицы A (результат выдается в виде линейной оболочки, порожденной вектор-столбцами).

Матрица F есть основная матрица системы (5.9) для нахождения неизвестных $c_1, c_2, ..., c_n$. Найдя ее единственное решение (переменная p), записываеМ искомый вектор оптимального приближения (вектор X_optimal).

3. Квадратичные формы

Основные задачи, рассматриваемые при изучении темы “Квадратичные формы”, связаны с приведением квадратичной формы (далее к.ф.) к каноническому (или нормальному) виду различными способами, с исследованием к.ф. на знакоопределенность. Рассматривается также задача о приведении пары к.ф. при помощи общего линейного преобразования к каноническому виду. Немаловажный момент относится и к нахождению соответствующего линейного преобразования (матрицы преобразования, канонического базиса).

В процессе изучения к.ф. одновременно с классическим подходом решения задач можно применять и современные системы компьютерной математики (далее СКМ), такие как Mathematica, Maple, MathCad, wxMaxima. Использование перечисленных СКМ позволяет проследить правильность проведенных расчетов, проверить полученный результат. Одновременно с этим при больших объемах данных (в случае к.ф. это означает достаточно большой порядок к.ф.) применение СКМ позволяет значительно уменьшить время проведения расчетов.

Рассмотрим использование СКМ Maple и wxMaxima при приведении к.ф. к каноническому виду (далее к.в.). Как правило, в этом случае

пользуются одним из трех способов: методом Лагранжа (выделения полных квадратов), методом ортогонального преобразования и методом Якоби.

Пусть к.ф. от переменных $x_1, x_2, ..., x_n$ записана в матричной форме

$$L(\bar{x}) = \bar{x}^T A \bar{x}, \qquad (5.10)$$

где $A = (a_{ij})_{i,j=1}^{n}$ - матрица формы ($a_{ij} = a_{ji}, i, j \in \{1, 2, ..., n\}$, $\bar{x} = (x_1 \quad ... \quad x_n)^T$.

Требуется привести к.ф. (1) к к.в.

$$L(\bar{y}) = \sum_{i=1}^{n} \mu_i y_i^2 = \bar{y}^T A' \bar{y}, \quad \mu_i \in \mathbf{R}, \qquad (5.11)$$

($A' = diag(\mu_1, \mu_2, ..., \mu_n)$, $\bar{y} = (y_1 \quad ... \quad y_n)^T$) и найти соответствующий диагонализирующий базис. Как известно, любую к.ф. (5.10) при помощи неособенного линейного преобразования $\bar{x} = T\bar{y}$ с матрицей T можно привести к к.в. (5.11).

Рассмотрим реализацию метода Лагранжа (последовательного выделения полных квадратов) в СКМ Maple. Для решения задачи помимо загрузки пакета `LinearAlgebra` используется пакет `Student[Precalculus]`, в котором есть процедура `Student[Precalculus][CompleteSquare]`, предназначенная для выделения полного квадрата в к.ф. при некоторой ведущей переменной. В результате последовательного выделения полных квадратов получим к.н. Для составления искомой матрицы T преобразования вводим дополнительные канонические переменные. Ниже приведен листинг данной программы в случае третьего порядка к.ф.

```
> restart; with(LinearAlgebra): with(Student[Precalculus]):
> A:=Matrix(3,3,[[a[1,1],a[1,2],a[1,3]],[a[1,2],a[2,2],a[2,3]],
  [a[1,3], a[2,3], a[3,3]]]):
   X:=Matrix(3,1,[[x[1]], [x[2]], [x[3]]]):
L:=a[1,1]*x[1]^2+a[2,2]*x[2]^2+a[3,3]*x[3]^2+2*a[1,2]*x[1]*x[2]+
   2*a[1,3]*x[1]*x[3]+2*a[2,3]*x[2]*x[3]:
> # задаем исходную квадратичную форму от трех переменных и
матрицу этой формы
a[1,1]:=1: a[2,2]:=5:a[3,3]:=-4:a[1,2]:=1:a[1,3]:=-2:a[2,3]:=0:
> # выделяем полный квадрат при x1
   L:=Student[Precalculus][Student:-Precalculus:-CompleteSquare]
(L,[x[1]]);
```

$$L := (x_1 - 2x_3 + x_2)^2 + 5x_2^2 - 4x_3^2 - \frac{1}{4}(-4x_3 + 2x_2)^2$$

```
> # выделяем полный квадрат при x2
L:=(x[1]-2*x[3]+x[2])^2+Student[Precalculus][Student:-
Precalculus:-CompleteSquare](4*x[2]^2-
8*x[3]^2+4*x[3]*x[2],[x[2]]);
```

$$L := (x_1 - 2x_3 + x_2)^2 + 4\left(x_2 + \frac{1}{2}x_3\right)^2 - 9x_3^2$$

```
> # вводим канонические переменные
y[1]:=x[1]-2*x[3]+x[2]: y[2]:=x[2]+x[3]/2: y[3]:=x[3]:
```

```
> # формируем матрицу S из коэффициентов при старых переменных
for i from 1 by 1 to 3 do
  for j from 1 by 1 to 3 do
    s[i,j]:=diff(y[i],x[j]):
  end do
end do;
  S:=Matrix(3,3,[[s[1,1],s[1,2],s[1,3]],[s[2,1],s[2,2],s[2,3]],
  [s[3,1],s[3,2],s[3,3]]]);
```

$$S := \begin{bmatrix} 1 & 1 & -2 \\ 0 & 1 & \frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

```
> # Находим матрицу преобразования, приводящего к.ф. к к.в., и
матрицу к.ф. канонического вида
P:=MatrixInverse(S);  Matrix_Canonical_Form:=MatrixMatrixMultiply
(MatrixMatrixMultiply(Transpose(P),A), P);
```

$$P := \begin{bmatrix} 1 & -1 & \frac{5}{2} \\ 0 & 1 & \frac{-1}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

$$Matrix\_Canonical\_Form := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & -9 \end{bmatrix}$$

Рассмотрим второй способ приведения к.ф. к к.в. – метод ортогональных преобразований. Данный метод основан на том, что для любой симметрической матрицы $A=\left(a_{ij}\right)_{i,j=1}^{n}$ существует ортогональная матрица $U=\left(u_{ij}\right)_{i,j=1}^{n}$ $(U^T\cdot U=E)$ такая, что выполняется равенство $A'=U^{\mathrm{T}}\cdot A\cdot U$, $A'=\operatorname{diag}\left(\lambda_1,\lambda_2,...,\lambda_n\right)$, где $\lambda_1,\lambda_2,...,\lambda_n\in\mathbf{R}$ – собственные значения матрицы A, повторяющиеся с учетом их алгебраических кратностей. При этом собственные векторы матрицы A, отвечающие различным собственным значениям, ортогональны (собственные векторы, соответствующие одному и тому же собственному значению не ортогональны, а значит, в этом случае необходимо проводить процесс ортогонализации Грамма - Шмидта). В результате получаем ортогональное преобразование

$$\overline{x}=U\cdot\overline{y}. \qquad (5.12)$$

Если в качестве матрицы A взять матрицу к.ф. (5.10), то при помощи преобразования (5.11) ее можно привести к диагональному виду $A'=\operatorname{diag}\left(\lambda_1,\lambda_2,...,\lambda_n\right)$. А это означает, что любую к.ф. (5.10) с помощью ортогонального преобразования (5.12) можно привести к следующему каноническому виду:

$$K\left(y_1,y_2,...,y_n\right)=\sum_{i=1}^{n}\lambda_i y_i^2. \qquad (5.13)$$

Рассмотрим реализацию метода ортогональных преобразований в СКМ

Maple. В программе с помощью процедуры Eigenvectors вычисляются собственные значения и соответствующие собственные векторы матрицы к.ф. Используя процедуру GramSchmidt, проводим процесс ортогонализации Грамма - Шмидта. Столбцами матрицы U являются ортонормированные собственные векторы. В результате матрица K есть диагональная матрица к.ф. к.в.

```
> restart; with(LinearAlgebra): n:=3:
>  A:=Matrix(3,3,[[a[1,1], a[1,2], a[1,3]], [a[1,2], a[2,2],
a[2,3]], [a[1,3], a[2,3], a[3,3]]]):
X:=Matrix(3,1,[[x[1]], [x[2]], [x[3]]]):
L:=a[1,1]*x[1]^2+a[2,2]*x[2]^2+a[3,3]*x[3]^2+2*a[1,2]*x[1]*x[2]+
2*a[1,3]*x[1]*x[3]+2*a[2,3]*x[2]*x[3]:
>  a[1,1]:=2:  a[2,2]:=5:  a[3,3]:=5:  a[1,2]:=2:  a[1,3]:=-2:
a[2,3]:=-4:
> # вычисляем собственные числа и соответственные собственные
векторы матрицы A квадратичной формы
  F:=Eigenvectors(A):
> Matrix_Eigen_Vectors:=Transpose(F[2]):
  v[1]:=Matrix_Eigen_Vectors[1]: v[2]:=Matrix_Eigen_Vectors[2]:
  v[3]:=Matrix_Eigen_Vectors[3]:
> # процесс ортогонализации системы собственных векторов
  Ortogonal_System:=GramSchmidt([v[1],v[2],v[3]]):
>  # нормируем собственные векторы и получаем систему
ортонормированных векторов
  u[1]:=Ortogonal_System[1]/Norm(Ortogonal_System[1],
Euclidean):
  u[2]:=Ortogonal_System[2]/Norm(Ortogonal_System[2],
Euclidean):
  u[3]:=Ortogonal_System[3]/Norm(Ortogonal_System[3],
Euclidean):
> # формируем матрицу U ортогонального преобразования
  U:=Transpose(Matrix(3,3,[[u[1]],[u[2]],[u[3]]])):
> # диагональная матрица к.ф. к.в.
  K:=Transpose(U).A.U:
> # вывод результатов
  L:=L; [F, U, K];
```

$$L := 2x_1^2 + 5x_2^2 + 5x_3^2 + 4x_1x_2 - 4x_1x_3 - 8x_2x_3$$

$$\left[\left[\begin{bmatrix}1\\1\\10\end{bmatrix},\begin{bmatrix}2 & -2 & \frac{-1}{2}\\0 & 1 & -1\\1 & 0 & 1\end{bmatrix},\begin{bmatrix}\frac{2\sqrt{5}}{5} & -\frac{2\sqrt{5}}{15} & \frac{-1}{3}\\0 & \frac{\sqrt{5}}{3} & \frac{-2}{3}\\\frac{\sqrt{5}}{5} & \frac{4\sqrt{5}}{15} & \frac{2}{3}\end{bmatrix},\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 10\end{bmatrix}\right]\right]$$

Рассмотрим реализацию метода ортогональных преобразований в СКМ wxMaxima. Процедура eigenvectors(A) выводит результат в виде списков. Первый список состоит из собственных значений матрицы к.ф. с указанием их алгебраических кратностей, второй – из соответствующих собственных векторов матрицы к.ф. При этом список, отвечающий собственным векторам матрицы, разделен на два списка в соответствии с собственными значениями.

При помощи процедуры `uniteigenvectors(A)` вычиляются нормированные собственные векторы матрицы к.ф. и проводится процесс ортогонализации Грамма - Шмидта для второго и третьего собственных векторов (встроенная процедура `gramshcmidt, присутствующая в СКМ` wxMaxima, не дает нужного результата относительно процесса ортогонализации). В результате выполнения команды `U:transpose(matrix(EigenVect1, EigenVect2, e))` выводится матрица U соответствующего ортогонального преобразования.

```
(%i37)  A: matrix([2,2,-2], [2,5,-4], [-2,-4,5]);
        x: matrix([x1,x2,x3]); KF: expand(x.A.transpose(x));
```

$$5\,x3^2-8\,x2\,x3-4\,x1\,x3+5\,x2^2+4\,x1\,x2+2\,x1^2 \quad \text{(KF)}$$

```
(%i21)  eigenvectors(A);
```

$$[[[10,1],[1,2]],[[[1,2,-2]],[[1,0,\frac{1}{2}],[0,1,1]]]] \quad \text{(\%o21)}$$

```
(%i41)  lambda[1]: uniteigenvectors(A)[1][1][1];
        lambda[2]: uniteigenvectors(A)[1][1][2];
(%i26)  EigenVect1: uniteigenvectors(A)[2][1][1];
        EigenVect2: uniteigenvectors(A)[2][2][1];
        EigenVect3: uniteigenvectors(A)[2][2][2];
```

$$[\frac{1}{3},\frac{2}{3},-\frac{2}{3}] \quad \text{(EigenVect1)}$$

$$[\frac{2}{\sqrt{5}},0,\frac{1}{\sqrt{5}}] \quad \text{(EigenVect2)}$$

$$[0,\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}] \quad \text{(EigenVect3)}$$

```
(%i29)  EigenVect1.EigenVect2; EigenVect1.EigenVect3;
        EigenVect2.EigenVect3;
(%i32)  alpha: EigenVect3.EigenVect2; g: EigenVect3-alpha.EigenVect2;
        e:g/sqrt(g.g);
```

$$[-\frac{\sqrt{2}}{5},\frac{1}{\sqrt{2}},\frac{2^{3/2}}{5}] \quad \text{(g)}$$

$$[-\frac{\sqrt{2}\sqrt{10}}{15},\frac{\sqrt{10}}{3\sqrt{2}},\frac{2^{3/2}\sqrt{10}}{15}] \quad \text{(e)}$$

```
(%i38)  U:transpose(matrix(EigenVect1, EigenVect2, e));
```

$$\begin{bmatrix} \frac{1}{3} & \frac{2}{\sqrt{5}} & -\frac{\sqrt{2}\sqrt{10}}{15} \\ \frac{2}{3} & 0 & \frac{\sqrt{10}}{3\sqrt{2}} \\ -\frac{2}{3} & \frac{1}{\sqrt{5}} & \frac{2^{3/2}\sqrt{10}}{15} \end{bmatrix} \quad \text{(U)}$$

Метод Якоби приведения к.ф. к к.в. основан на использовании угловых миноров $\Delta_k = \det A^{(k)}$ $(A^{(k)}_{k\times k} = \left(a_{ij}\right)_{i,j=1}^{k},\ k=\overline{1,n})$ матрицы $A=\left(a_{ij}\right)_{i,j=1}^{n}$ к.ф. Достаточным условием приведения к.ф. к к.в. методом Якоби является выполнение условия Якоби, при котором все угловые миноры матрицы

$A=\left(a_{ij}\right)_{i,\,j=1}^{n}$ (в некотором базисе), кроме, быть может, последнего $\Delta_n=\det A$, должны быть отличны от нуля. При этом условии существует базис, в котором матрица имеет диагональный вид $A'=diag\left\{\mu_1,\ \mu_2,\ ...,\ \mu_n\right\},\ \mu_i=\dfrac{\Delta_i}{\Delta_{i-1}}$ ($i=\overline{1,\ n},\ \Delta_0=1$), а к.ф. имеет к.в.

$$K\left(y_1,\ y_2,\ ...,\ y_n\right)=\sum_{i=1}^{n}\mu_i y_i^2.$$

Пусть для определенности n=4. Тогда матрица T линейного преобразования, при котором матрица $A'=T^T\cdot A\cdot T$ является диагональной, строится в верхнетреугольном виде

$$T=\begin{pmatrix}1 & t_{12} & t_{13} & t_{14}\\ 0 & 1 & t_{23} & t_{24}\\ 0 & 0 & 1 & t_{34}\\ 0 & 0 & 0 & 1\end{pmatrix},$$

где t_{ij} – неизвестные числа ($1\le i<j\le 4$). При этом элементы матрицы T однозначно определяются из квадратной системы линейных алгебраических уравнений $A\cdot\tilde{t}=\tilde{b}$ с неособенной квадратной матрицей $\underset{6\times6}{A}$:

$$\underset{6\times6}{A}=diag\left\{A^{(1)},\ A^{(2)},\ A^{(3)}\right\}=\begin{pmatrix}\boxed{a_{11}} & 0 & 0 & 0 & 0 & 0\\ 0 & \overline{\boxed{a_{11}}} & \overline{\boxed{a_{12}}} & 0 & 0 & 0\\ 0 & \underline{\boxed{a_{21}}} & \underline{\boxed{a_{22}}} & 0 & 0 & 0\\ 0 & 0 & 0 & \overline{\boxed{a_{11}}} & a_{12} & \overline{\boxed{a_{13}}}\\ 0 & 0 & 0 & a_{21} & a_{22} & a_{23}\\ 0 & 0 & 0 & \underline{\boxed{a_{31}}} & a_{32} & \underline{\boxed{a_{33}}}\end{pmatrix},$$

$$\tilde{t}=\begin{pmatrix}t_{12}\\ t_{13}\\ t_{23}\\ t_{14}\\ t_{24}\\ t_{34}\end{pmatrix},\ \tilde{b}=\begin{pmatrix}-a_{12}\\ -a_{13}\\ -a_{23}\\ -a_{14}\\ -a_{24}\\ -a_{34}\end{pmatrix},$$

$\det A=\det A^{(1)}\cdot\det A^{(2)}\cdot\det A^{(3)}=\Delta_1\cdot\Delta_2\cdot\Delta_3\neq 0$.

Рассмотрим реализацию метода Якоби в СКМ Maple (при n=4). В процессе решения задачи в цикле `for` идет проверка выполнимости условия Якоби с выводом соответствующего сообщения. В качестве результата выводятся матрица T преобразования Якоби, диагональная матрица A' и к.ф. к.в.

```
> restart; with(LinearAlgebra): n:=4:
> X:=x[1]: Y:=y[1]:
for i from 2 by 1 to n do
  X:=<X|x[i]>; Y:=<Y|y[i]>;
end do:
> X:=Transpose(X): Y:=Transpose(Y): #формируем вектор-столбцы
> a[1,1]:=12: a[2,2]:=10: a[3,3]:=2: a[4,4]:=1: a[1,2]:=0:
  a[1,3]:=0: a[3,4]:=-1: a[2,3]:=4: a[2,4]:=0: a[1,4]:=-2:
> for i from 1 by 1 to n do
   for j from i+1 by 1 to n do
    a[j,i]:=a[i,j]:
   end do;
  end do; #формируем элементы симметрической матрицы A формы
> A:=Matrix(n,n, a): r:=Rank(A):
  L:=simplify(Transpose(X).A.X)[1,1];
```

$$L := 12\,x_1^{2} - 4\,x_1 x_4 + 10\,x_2^{2} + 8\,x_2 x_3 + 2\,x_3^{2} - 2\,x_3 x_4 + x_4^{2}$$

```
> Delta[0]:=1:
> for i from 1 by 1 to n do
  _A[i]:=SubMatrix(A, [1..i],[1..i]):
  Delta[i]:=Determinant(_A[i]):   #вычисляем   угловые   миноры
матрицы A
  if i<n and Delta[i]=0 then
  #  Если  очередной  (не  последний)  угловой  минор  оказывается
   нулевым, то выдается сообщение об ошибке
  ERROR("Условия Якоби не выполнены!");
  end if;
end do:
print(`Условия Якоби для квадратичной формы выполнены`);
print(`Метод Якоби для квадратичной формы применим`);
```

Условия Якоби для квадратичной формы выполнены

Метод Якоби для квадратичной формы применим

```
> #формируем блочную матрицу - основную матрицу системы для
нахождения наддиагональных элементов в матрице T
Block_Matrix:=DiagonalMatrix([_A[1], _A[2], _A[3]]):
> #формируем вектор-столбец коэффициентов системы для нахождения
наддиагональных элементов в матрице T
b:=Vector(6,[-a[1,2],-a[1,3],-a[2,3],-a[1,4],-a[2,4],-a[3,4]]):
> #  решение  СЛАУ  относительно  наддиагональных  элементов  в
матрице T
  c:=Transpose(LinearSolve(Block_Matrix, b)):
> k:=1: # заполнение элементов матрицы T (элементы выше главной
диагонали пока не найдены)
for i from 1 by 1 to n do
 for j from 1 by 1 to n do
   if (i=j) then T[i,j]:=1 end if:
   if (i>j) then T[i,j]:=0 end if:
 end do;
end do;
> #заполняем наддиагональные элементы матрицы T
T[1,2]:=c[1]:    T[1,3]:=c[2]:    T[2,3]:=c[3]:    T[1,4]:=c[4]:
T[2,4]:=c[5]: T[3,4]:=c[6]:
T:=Matrix(n,n,T):
> # вывод результатов
```

```
_A_:=Transpose(T).A.T: K:=simplify(Transpose(Y)._A_.Y)[1,1]:
[T, _A_, K];
```

$$\left[\begin{bmatrix} 1 & 0 & 0 & \frac{1}{6} \\ 0 & 1 & \frac{-2}{5} & -1 \\ 0 & 0 & 1 & \frac{5}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 12 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & \frac{2}{5} & 0 \\ 0 & 0 & 0 & \frac{-11}{6} \end{bmatrix}, 12\,y_1^{\,2} + 10\,y_2^{\,2} + \frac{2}{5}y_3^{\,2} - \frac{11}{6}y_4^{\,2}\right]$$

Еще одной из классических задач линейной алгебры является исследование к.ф. на знакоопределенность. Тип к.ф. можно определить, приведя ее к каноническому виду (например, методом Якоби). Исследуем на знакоопределенность в зависимости от значения параметра $\lambda \in \mathbf{R}$ для к.ф. от переменных x_1, x_2, x_3:

$$L(x_1, x_2, x_3) = (\lambda - 1)x_1^2 + x_2^2 + (\lambda + 1)x_3^2 - 2x_1x_2 + 4x_1x_3.$$

Исследование проводится следующим образом. Вычисляются угловые миноры $\Delta_i(\lambda)$ $(i = 1,2,3)$ матрицы к.ф. и определяется множество Λ значений λ, при которых ни один из угловых миноров $\Delta_i(\lambda)$ $(i = 1,2,3)$ не обращается в нуль. На множестве Λ применим метод Якоби – к.ф. примет к.в.

$$K(y_1, y_2, y_3) = \sum_{i=1}^{3} \mu_i(\lambda) y_i^2 = \sum_{i=1}^{3} \frac{\Delta_i(\lambda)}{\Delta_{i-1}(\lambda)} y_i^2$$

с каноническими коэффициентами $\mu_i(\lambda) = \frac{\Delta_i(\lambda)}{\Delta_{i-1}(\lambda)}$ $(i = 1,2,3)$.

Далее тип к.ф. определяется в зависимости от знаков канонических коэффициентов $\mu_i(\lambda)$. Если при всех $i = 1,2,3$ на некотором множестве $\Lambda_1 \subset \Lambda$: $\mu_i(\lambda) > 0$, то на Λ_1 к.ф. является положительно определенной. Если при всех $i = 1,2,3$ на множестве $\Lambda_2 \subset \Lambda$: $\mu_i(\lambda) < 0$, то на Λ_2 к.ф. является отрицательно определенной. На множестве $\Lambda \setminus (\Lambda_1 \cup \Lambda_2)$ к.ф. является знакопеременной. При значениях $\lambda \notin \Lambda$ (то есть при тех значениях параметра, при которых какой-то канонический коэффициент $\mu_i(\lambda)$ обращается в нуль) исследование проводится непосредственно.

```
> restart; with(LinearAlgebra): n:=3:
>  a[1,1]:=lambda-1:  a[2,2]:=1:  a[3,3]:=lambda+1:  a[1,2]:=-1:
a[1,3]:=2: a[2,3]:=0:
# вводим элементы матрицы к.ф.
> for i from 1 by 1 to n do
   for j from i+1 by 1 to n do
    a[j,i]:=a[i,j]:
   end do;
  end do; #формируем элементы симметрической матрицы А формы
>A:=Matrix(n,n, a):
> Kanon_Forma:=proc()
  # процедура построения к.ф.
  global i, n, mu, K, y;
```

```
  K:=0:
  for i from 1 by 1 to n do
    mu[i]:=Delta[i]/Delta[i-1]: K:=K+mu[i]*y[i]^2:
  end do:
  RETURN(K);
  end proc:
> Proverka_Uslovy_Yakobi:=proc()
  # процедура проверки условий Якоби для матрицы A к.ф.
   global Delta, n, i, B, _B;
   Delta[0]:=1:
   for i from 1 by 1 to n do
     _B[i]:=SubMatrix(B, [1..i],[1..i]):
     Delta[i]:=Determinant(_B[i]):
   #вычисляем угловые миноры матрицы B
     if i<n and Delta[i]=0 then # Если очередной (не последний)
угловой минор оказывается нулевым, то выдается сообщение об
ошибке
     ERROR("Usloviya Yakobi ne vypolneny!");
     end if;
   end do;
   RETURN('Matrica_udovletvoryaet_usloviyam_Yakobi');
end proc:
> B:=A: Kanon_Forma();
```

$$(\lambda-1)\,y_1^{\,2}+\frac{(\lambda-2)\,y_2^{\,2}}{\lambda-1}+\frac{(\lambda^2-6-\lambda)\,y_3^{\,2}}{\lambda-2}$$

```
> for i from 1 by 1 to n do
  Null_Delta[i]:=solve(Delta[i], lambda);
end do;
```

$$Null\_Delta_1 := 1$$

$$Null\_Delta_2 := 2$$

$$Null\_Delta_3 := 3, -2$$

```
>   Interval_Pol_opredelen:=solve([mu[1]>0,   mu[2]>0,   mu[3]>0],
lambda);
```

$$Interval\_Pol\_opredelen := \{3 < \lambda\}$$

```
>   Interval_Otr_opredelen:=solve([mu[1]<0,   mu[2]<0,   mu[3]<0],
lambda);
```

$$Interval\_Otr\_opredelen :=$$

```
> B:=A: lambda:=Null_Delta[1]; B; Proverka_Uslovy_Yakobi();
```

$$\lambda := 1$$

$$\begin{bmatrix} 0 & -1 & 2 \\ -1 & 1 & 0 \\ 2 & 0 & 2 \end{bmatrix}$$

```
Error, (in Proverka_Uslovy_Yakobi) Usloviya Yakobi ne vypolneny!
>      B:=ColumnOperation(B,[1,2]):      B:=RowOperation(B,[1,2]);
Proverka_Uslovy_Yakobi();
```

$$B := \begin{bmatrix} 1 & -1 & 0 \\ -1 & 0 & 2 \\ 0 & 2 & 2 \end{bmatrix}$$

$$Matrica\_udovletvoryaet\_usloviyan\_Yakobi$$

```
> Kanon_Forma();
```

$$y_1^2 - y_2^2 + 6y_3^2$$

```
> lambda:='lambda':
> B:=A: lambda:=Null_Delta[2]; B; Proverka_Uslovy_Yakobi();
```

$$\lambda := 2$$

$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 1 & 0 \\ 2 & 0 & 3 \end{bmatrix}$$

Error, (in Proverka_Uslovy_Yakobi) Usloviya Yakobi ne vypolneny!

```
>      B:=ColumnOperation(B,[2,3]):      B:=RowOperation(B,[2,3]);
Proverka_Uslovy_Yakobi();
```

$$B := \begin{bmatrix} 1 & 2 & -1 \\ 2 & 3 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

Matrica_udovletvoryaet_usloviyan_Yakobi

```
> Kanon_Forma();
```

$$y_1^2 - y_2^2 + 4y_3^2$$

```
> lambda:='lambda':
> B:=A: lambda:=Null_Delta[3][1]; B; Proverka_Uslovy_Yakobi();
```

$$\lambda := 3$$

$$\begin{bmatrix} 2 & -1 & 2 \\ -1 & 1 & 0 \\ 2 & 0 & 4 \end{bmatrix}$$

Matrica_udovletvoryaet_usloviyan_Yakobi

```
> Kanon_Forma();
```

$$2y_1^2 + \frac{1}{2}y_2^2$$

```
> lambda:='lambda':
> B:=A: lambda:=Null_Delta[3][2]; B; Proverka_Uslovy_Yakobi();
```

$$\lambda := -2$$

$$\begin{bmatrix} -3 & -1 & 2 \\ -1 & 1 & 0 \\ 2 & 0 & -1 \end{bmatrix}$$

Matrica_udovletvoryaet_usloviyan_Yakobi

```
> Kanon_Forma();
```

$$-3y_1^2 + \frac{4}{3}y_2^2$$

В программе использованы две пользовательские процедуры `Kanon_Forma`, `Proverka_Uslovy_Yakobi`. Первая из них предназначена для построения канонической к.ф. методом Якоби, а вторая – для проверки условий Якоби для матрицы к.ф. Для определения интервалов положительной определенности (переменная `Interval_Pol_opredelen`) и отрицательной определенности (переменная `Interval_Otr_opredelen`) к.ф. решаются соответствующие неравенства. Далее исследование на знакоопределенность проводится при значениях $\lambda \notin \Lambda$ (в программе – переменная `Null_Delta`). При значении $\lambda = 1$ для матрицы к.ф. условия Якоби не

выполняются. Тогда, используя процедуры ColumnOperation, RowOperation, меняем местами первые две строки и первые два столбца матрицы формы. При этом полученная матрица будет удовлетворять условиям Якоби.

Рассмотрим далее одно из важных приложений к.ф. – задачу приведения пары к.ф. $L_1(\bar{x}) = \bar{x}^T A\bar{x}$, $L_2(\bar{x}) = \bar{x}^T B\bar{x}$ к каноническому виду общим неособенным линейным преобразованием.

Решение задачи основано на понятии симметрической билинейной формы (б.ф.) $f(\bar{x}, \bar{y})$ как функции от переменных $\bar{x}, \bar{y} \in \mathbf{R}^n$, линейной по каждому из своих аргументов $\bar{x}, \bar{y}$.

Если в $\mathbf{R}^n$ выбрать базис $\mathbf{B} = (\bar{f}_1, \bar{f}_2, ..., \bar{f}_n)$ и обозначить $a_{ij} = f(\bar{f}_i, \bar{f}_j)$ $(i, j = \overline{1,n})$, то для $\bar{x} = (x_1 \quad ... \quad x_n)^T$, $\bar{y} = (y_1 \quad ... \quad y_n)^T$:

$$f(\bar{x}, \bar{y}) = f\left(\sum_{i=1}^{n} x_i \bar{f}_i, \sum_{j=1}^{n} y_j \bar{f}_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} x_i y_j \cdot f(\bar{f}_i, \bar{f}_j) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i y_j = \bar{x}^T A\bar{y},$$

где $A = (a_{ij})_{i,j=1}^{n}$. Итак, б.ф. имеет матричную форму записи

$$f(\bar{x}, \bar{y}) = \bar{x}^T A\bar{y}.$$

При $\bar{x} = \bar{y}$, $A = A^T$ получим: $f(\bar{x}, \bar{x}) = \bar{x}^T A\bar{x} = L_1(\bar{x})$, то есть симметрическая б.ф. $f(\bar{x}, \bar{y})$ порождает к.ф. $L(\bar{x}) = \bar{x}^T A\bar{x}$.

Рассмотрим первый случай, когда $L_1(\bar{x})$ является положительно определенной к.ф. Тогда соответствующая ей симметрическая б.ф. $f_L(\bar{x}, \bar{y})$ задает в пространстве $\mathbf{R}^n$ скалярное произведение $(\bar{x}, \bar{y}) = f_L(\bar{x}, \bar{y}) = \bar{x}^T A\bar{y}$, и существует общий ортонормированный (диагонализирующий) базис $\mathbf{B}_N = (\bar{e}_1, \bar{e}_2, ..., \bar{e}_n)$, в котором к.ф. $L_1(\bar{x})$ и $L_2(\bar{x})$ имеет канонический вид:

$$K_1(\bar{y}) = \sum_{i=1}^{n} y_i^2, \quad K_2(\bar{y}) = \sum_{i=1}^{n} \lambda_i y_i^2, \qquad (5.14)$$

Решение задачи основано на следующем алгоритме:

1) рассмотреть линейный (сопровождающий) оператор $\mathbf{C}: \mathbf{R}^n \to \mathbf{R}^n$, заданный матрицей $A^{-1}B$, найти его собственные значения λ_i $(i = \overline{1,n})$ и соответствующие собственные векторы $\bar{f}_i$ $(i = \overline{1,n})$;

2) для системы векторов $\bar{f}_i$ $(i = \overline{1,n})$ провести процесс ортогонализации Грама - Шмидта относительно скалярного произведения $(\bar{x}, \bar{y}) = \bar{x}^T A\bar{y}$, в

результате чего получить ортонормированный базис $\mathbf{B}_N = \left(\overline{e_1},\ \overline{e_2},\ldots,\ \overline{e_n}\right)$, являющийся общим диагонализирующим базисом для пары к.ф.;

3) составить матрицу $\underset{n\times n}{U} = \left(\overline{e_1}\middle|\overline{e_2}\middle|\ldots\middle|\overline{e_n}\right)$ неособенного линейного преобразования $\overline{x} = U\overline{y},\ \overline{y} = \left(y_1 \quad y_2 \quad \ldots \quad y_n\right)^T$. В базисе $\mathbf{B}_N$ к.ф. $L_1\left(\overline{x}\right) = \overline{x}^T A\overline{x}$ будет иметь нормальный вид (5.14), к.ф. $L_2\left(\overline{x}\right) = \overline{x}^T B\overline{x}$ – канонический вид (5.14).

Рассмотрим пары к.ф.

$$L_1\left(\overline{x}\right) = x_1^2 + 2x_1x_2 + 3x_2^2,\ L_2\left(\overline{x}\right) = 4x_1^2 + 16x_1x_2 + 6x_2^2.$$

В фрагменте программы в СКА Maple идет проверка первой к.ф. на положительную определенность при помощи критерия Сильвестра (с выводом сообщения в том случае, когда к.ф. не является положительно определенной). Для матрицы $C = A^{-1}B$ находятся собственные значения и соответствующие собственные векторы. Для полученной системы собственных векторов проводится процесс ортогонализации Грамма-Шмидта относительно скалярного произведения $\left(\overline{x},\ \overline{y}\right) = \overline{x}^T A\overline{y}$, в результате которого получаем матрицу U преобразования, и матрицы к.ф. к.в.

```
> A:=Matrix(n,n, a): B:=Matrix(n,n, b):
L[1]:=simplify(Transpose(X).A.X);
L[2]:=simplify(Transpose(X).B.X);
```

$$L_1 := x_1^2 + 2\,x_1\,x_2 + 3\,x_2^2 \qquad L_2 := 4\,x_1^2 + 16\,x_1\,x_2 + 6\,x_2^2$$

```
> for i from 1 by 1 to n do
  Delta[i]:=Determinant(SubMatrix(A, [1..i],[1..i])):
  #вычисляем угловые миноры матрицы A
  if Delta[i]=0 or Delta[i]<0 then
  # Если очередной угловой минор неположительный, то выдается
  сообщение об ошибке
  ERROR("форма не является положительно определенной");
  end if:
end do:
> # выводим матрицы собственных значений и собственных векторов
матрицы C
C:=MatrixInverse(A).B: F:=Eigenvectors(C);
Matrix_Eigen_Vectors:=F[2]:
```

$$F := \begin{bmatrix} -4 \\ 5 \end{bmatrix}, \begin{bmatrix} \frac{-3}{2} & 3 \\ 1 & 1 \end{bmatrix}$$

```
> for i from 1 by 1 to n do
  f[i]:=Matrix_Eigen_Vectors[1,i]:
  for j from 2 by 1 to n do
    f[i]:=<f[i]|Matrix_Eigen_Vectors[j,i]>:
  end do;
f[i]:=Transpose(f[i]);
end do:
```

```
> # проводим процесс ортогонализации Грама - Шмидта относительно введенного скалярного произведения
> Norma[1]:=sqrt(Transpose(f[1]).A.f[1]):
  e[1]:=simplify(f[1]/Norma[1]):
  g[2]:=f[2]-(Transpose(f[2]).A.e[1]).e[1]:
  Norma[2]:=sqrt(Transpose(g[2]).A.g[2]):
  e[2]:=simplify(g[2]/Norma[2]):
> U:=e[1]:
  for i from 2 by 1 to n do
   U:=<U|e[i]>;
  end do:
U:=U;      #выводим матрицу U, столбцами которой являются ортонормированные векторы
```

$$U := \begin{bmatrix} -1 & \frac{\sqrt{2}}{2} \\ \frac{2}{3} & \frac{\sqrt{2}}{6} \end{bmatrix}$$

```
> _A_:=simplify(Transpose(U).A.U):K[1]:=simplify(Transpose(Y)._A_.Y):
  _B_:=simplify(Transpose(U).B.U):K[2]:=simplify(Transpose(Y)._B_.Y):
# выводим соответствующие канонические формы и их матрицы
[_A_, _B_]; [K[1], K[2]];
```

$$\left[\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} -4 & 0 \\ 0 & 5 \end{bmatrix}\right]$$

$$[y_1^2 + y_2^2, -4\,y_1^2 + 5\,y_2^2]$$

Ниже представлена реализация алгоритма приведения пары квадратичных форм к каноническому виду в СКА wxMaxima.

```
(%i3)  A:matrix([1,1],[1,3])$ B:matrix([4,8],[8,6])$
       C:invert(A).B$

(%i4)  D: eigenvectors(C);

(D)    [[[5,-4],[1,1]],[[[1,1/3]],[[1,-2/3]]]]

(%i16) lambda[1]:D[1][1][1]$ lambda[2]:D[1][1][2]$
       v[1]:D[2][1]$ v[2]:D[2][2]$

(%i18) F1:matrix([v[1][1][1], v[1][1][2]])$
       F2:matrix([v[2][1][1], v[2][1][2]])$
```

```
(%i43)  Norma1:sqrt(F1.A.transpose(F1))$
        E1:F1/Norma1$
        G2:F2-F2.A.transpose(E1)*E1$
        Norma2:sqrt(G2.A.transpose(G2))$
        E2:G2/Norma2$
        U:matrix([E1[1][1],E2[1][1]],[E1[1][2],E2[1][2]])$
        _A_:transpose(U).A.U$  _B_:transpose(U).B.U$
        Matr_KF:[U,_A_,_B_];
```

$$(\text{Matr\_KF})\quad \left[\begin{pmatrix} \frac{1}{\sqrt{2}} & 1 \\ \frac{1}{3\sqrt{2}} & -\frac{2}{3} \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 5 & 0 \\ 0 & -4 \end{pmatrix}\right]$$

Рассмотрим второй случай, когда к.ф. $L_1\left(\bar{x}\right)=\bar{x}^T A\bar{x}$ является невырожденной ($\det A\neq 0$). Как и выше (см. первый случай), к.ф. порождают симметрические б.ф. $f\left(\bar{x},\bar{y}\right)=\bar{x}^T A\bar{y}$, $g\left(\bar{x},\bar{y}\right)=\bar{x}^T B\bar{y}$. Дальнейшее решение задачи основано на следующей теореме.

Теорема 1 (достаточное условие существования общего диагонализирующего базиса для пары к.ф.) [11]. Если сопровождающий оператор $\mathbf{C}\colon \mathbf{R}^n\to\mathbf{R}^n$, заданный матрицей $A^{-1}B$, диагонализируем, то существует общий диагонализирующий базис $\mathbf{B}_N$, в котором к.ф. $L_1\left(\bar{x}\right)$, $L_2\left(\bar{x}\right)$ имеют канонический вид.

Доказательство теоремы опирается на использование собственных подпространств сопровождающего оператора пары к.ф. и на теореме о том, что матрица линейного оператора диагонализируема, если сумма размерностей всех собственных подпространств оператора равна размерности всего пространства (в нашем случае пространства $\mathbf{R}^n$):

$$\sum_{i=1}^{m}\dim S\left(\lambda_i\right)=\dim V=n\ \left(m\leq n\right). \qquad (5.15)$$

На основании теоремы можно составить алгоритм приведения пары к.ф. $L_1\left(\bar{x}\right)=\bar{x}^T A\bar{x}$, $L_2\left(\bar{x}\right)=\bar{x}^T B\bar{x}$ к каноническому виду при помощи общего невырожденного линейного преобразования [к.ф. $L_1\left(\bar{x}\right)$ является невырожденной]:

1) найти собственные значения $\lambda_i\ \left(i=\overline{1,m},\ m\leq n\right)$ (с учетом алгебраических кратностей) сопровождающего оператора $\mathbf{C}\colon \mathbf{R}^n\to\mathbf{R}^n$ и получить спектр Λ сопровождающего оператора;

2) для каждого собственного значения $\lambda_i\ \left(i=\overline{1,m},\ m\leq n\right)$ построить собственное подпространство $S\left(\lambda_i\right)$ сопровождающего оператора, найти его размерность $\dim S\left(\lambda_i\right)$, проверить справедливость равенства (5.15). Если

(5.15) выполняется, то для пары к.ф. существует общий диагонализирующий базис $\mathbf{B}_N$;

3) в случае если спектр Λ сопровождающего оператора простой:

$\Lambda = \{\lambda_1,\ \lambda_2,\ ...,\ \lambda_n\},\ \lambda_i \neq \lambda_j\ \left(i \neq j,\ i, j = \overline{1,n}\right)$,

то общий диагонализирующий базис $\mathbf{B}_N = \left(\overline{e_1},\ \overline{e_2},...,\ \overline{e_n}\right)$, где $\overline{e_i}\ \left(i = \overline{1,n}\right)$ есть собственный вектор, соответствующий собственному значению $\lambda_i\ \left(i = \overline{1,n}\right)$;

4) пусть спектр Λ сложный и имеет, например, вид:

$\Lambda = \{\lambda,\ \lambda_1,\ \lambda_2,\ ...,\ \lambda_{n-k}\},\ \lambda_i \neq \lambda_j\ \left(i \neq j,\ i, j = \overline{1,n-k}\right)$,

где собственное значение λ имеет геометрическую кратность $k > 1$. В этом случае $\dim S(\lambda) = k$ и собственное подпространство $S(\lambda)$ порождено линейно независимой системой векторов $\overline{v_1},\ \overline{v_2},\ ...,\ \overline{v_k} \in S(\lambda)$.

Так как $\overline{v_1},\ \overline{v_2},\ ...,\ \overline{v_k} \in S(\lambda)$, то они не ортогональны относительно симметрической б.ф. $f\left(\overline{x}, \overline{y}\right)$. В этом случае систему векторов $\overline{v_1},\ \overline{v_2},\ ...,\ \overline{v_k}$ можно ортогонализировать, применяя процесс ортогонализации Грама - Шмидта, заменяя скалярное произведение значениями симметрической б.ф. $f\left(\overline{x}, \overline{y}\right)$:

$$
\begin{cases}
\overline{g_1} = \overline{v_1},\ \overline{g_1} \in S(\lambda), \\
\overline{g_2} = \overline{v_2} - \dfrac{f\left(\overline{g_1},\ \overline{v_2}\right)}{f\left(\overline{g_1},\ \overline{g_1}\right)} \cdot \overline{g_1},\ \overline{g_2} \in S(\lambda), \\
\overline{g_3} = \overline{v_3} - \dfrac{f\left(\overline{g_1},\ \overline{v_3}\right)}{f\left(\overline{g_1},\ \overline{g_1}\right)} \cdot \overline{g_1} - \dfrac{f\left(\overline{g_2},\ \overline{v_3}\right)}{f\left(\overline{g_2},\ \overline{g_2}\right)} \cdot \overline{g_2},\ \overline{g_3} \in S(\lambda), \\
\text{.....} \\
\overline{g_k} = \overline{v_k} - \dfrac{f\left(\overline{g_1},\ \overline{v_k}\right)}{f\left(\overline{g_1},\ \overline{g_1}\right)} \cdot g_1 - ... - \dfrac{f\left(\overline{g_{k-1}},\ \overline{v_k}\right)}{f\left(\overline{g_{k-1}},\ \overline{g_{k-1}}\right)} \cdot g_{k-1},\ \overline{g_k} \in S(\lambda),
\end{cases}
$$

причем $f\left(\overline{g_i},\ \overline{g_j}\right) = 0$ при всех $i, j = \overline{1,k},\ i \neq j$.

В результате общий канонический базис для пары к.ф. будет иметь вид

$\mathbf{B}_N = \left(\overline{e_1},\ \overline{e_2},...,\ \overline{e_k},\ \overline{e_{k+1}},\ \overline{e_{k+2}},...,\ \overline{e_n}\right) = \left(\overline{g_1},\ \overline{g_2},\ ...,\ \overline{g_k},\ \overline{v_{k+1}},\ \overline{v_{k+2}},\ ...,\ \overline{v_n}\right)$;

5) составить матрицу $U = \left(\overline{e_1} \middle| \overline{e_2} \middle| ... \middle| \overline{e_n}\right)$, столбцами которой являются векторы базиса $\mathbf{B}_N$, являющуюся матрицей искомого неособенного линейного преобразования

$\overline{x} = U\overline{y},\ \overline{y} = \left(y_1 \quad y_2 \quad ... \quad y_n\right)^T$.

В базисе $\mathbf{B}_N$ к.ф. $L_1\left(\bar{x}\right)$, $L_2\left(\bar{x}\right)$ имеют соответственно канонический вид

$$K_1\left(\bar{y}\right)=\sum_{i=1}^{n} a_{ii}'y_i^2,\ K_2\left(\bar{y}\right)=\sum_{i=1}^{n}\lambda_i a_{ii}'y_i^2,$$

где $a_{ii}'=\bar{e}_i^T A\bar{e}_i=L_1\left(\bar{e}_i\right),\ \left(i=\overline{1,n}\right)$.

Фрагмент программы в СКА Maple приведения пары к.ф.
$L_1\left(\bar{x}\right)=2x_1^2+2x_1x_2+x_2^2-2x_1x_3-2x_3^2,\ L_2\left(\bar{x}\right)=2x_1^2+2x_1x_2-2x_1x_3+4x_2x_3-6x_3^2$
при помощи общего невырожденного линейного преобразования приведен ниже. Для матрицы $C=A^{-1}B$ находятся собственные значения и собственные векторы. Для полученной системы собственных векторов матрицы $C=A^{-1}B$ проводится процесс ортогонализации относительно симметрической билинейной формы $f\left(\bar{x},\bar{y}\right)$, в результате которого получаем матрицу U преобразования и матрицы к.ф. к.в.

```
> C:=MatrixInverse(A).B:
> F:=Eigenvectors(C); Matrix_Eigen_Vectors:=F[2]:
```

$$F:=\begin{bmatrix}1\\1\\2\end{bmatrix},\begin{bmatrix}0&1&0\\2&0&1\\1&0&1\end{bmatrix}$$

```
> for i from 1 by 1 to n do
  v[i]:=Matrix_Eigen_Vectors[1,i]:
  for j from 2 by 1 to n do
    v[i]:=<v[i]|Matrix_Eigen_Vectors[j,i]>:
  end do;
v[i]:=Transpose(v[i]);
end do:
> for i from 1 by 1 to n do
  v[i]:=v[i]:
end do;
```

$$v_1:=\begin{bmatrix}0\\2\\1\end{bmatrix}\quad v_2:=\begin{bmatrix}1\\0\\0\end{bmatrix}\quad v_3:=\begin{bmatrix}0\\1\\1\end{bmatrix}$$

```
> # проверяем ортогональность собственных векторов относительно
СБФ A(x,y)
> Transpose(v[1]).A.v[2]; Transpose(v[1]).A.v[3];
 Transpose(v[2]).A.v[3];
```

$$\begin{bmatrix}0\\0\\1\end{bmatrix}$$

```
> # проводим процесс ортогонализации для векторов v[2], v[3]
относительно СБФ A(x,y)
> g[1]:=v[2]:
> p:=Transpose(g[1]).A.v[3]: q:=Transpose(g[1]).A.g[1]:
 alpha:=-p[1,1]/q[1,1]:
> g[2]:=v[3]+alpha*g[1]:
> e[1]:=v[1]: e[2]:=g[1]: e[3]:=g[2]:
```

```
> U:=Matrix(3,3,[e[1],e[2],e[3]]);
 #общий канонический базис для пары к.ф.
```

$$U := \begin{bmatrix} 0 & 0 & 1 \\ 1 & 2 & -1 \\ 1 & 1 & \frac{-1}{2} \end{bmatrix}$$

```
>
_A_:=simplify(Transpose(U).A.U):_B_:=simplify(Transpose(U).B.U):
K[1]:=simplify(Transpose(Y)._A_.Y)[1,1]:
K[2]:=simplify(Transpose(Y)._B_.Y)[1,1]:
# диагональные матрицы к.ф. и канонические к.ф. в общем каноническом базисе
[_A_,_B_]; [K[1],K[2]];
```

$$\_A\_ := \begin{bmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{3}{2} \end{bmatrix} \quad \_B\_ := \begin{bmatrix} -2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{3}{2} \end{bmatrix}$$

$$K_1 := -y_1^{\,2} + 2y_2^{\,2} + \frac{3}{2}y_3^{\,2} \quad K_2 := -2y_1^{\,2} + 2y_2^{\,2} + \frac{3}{2}y_3^{\,2}$$

Ниже представлена реализация алгоритма приведения пары квадратичных форм к каноническому виду в СКА wxMaxima.

```
(%i3)   A:matrix([2,1,-1],[1,1,0],[-1,0,-2])$
        B:matrix([2,1,-1],[1,0,2],[-1,2,-6])$
        C:invert(A).B$

(%i5)   D: eigenvectors(C);

(D)     [[[1,2],[2,1]],[[[1,0,0],[0,1,1/2]],[[0,1,1]]]]

(%i12)  v[1]:D[2][1][1]$ v[2]:D[2][1][2]$
        v[3]:D[2][2][1]$ [v[1], v[2],v[3]];
        Dot_Product12: v[1].A.transpose(v[2]);
        Dot_Product13: v[1].A.transpose(v[3]);
        Dot_Product23: v[2].A.transpose(v[3]);

(%o9)   [[1,0,0],[0,1,1/2],[0,1,1]]

(Dot_Product12) 1/2
(Dot_Product13) 0
(Dot_Product23) 0
```

```
(%i18)  g1:v[1]$
        g2:v[2]-(g1.A.transpose(v[2])/g1.A.transpose(g1))*g1$
        Dot_Product12:g1.A.transpose(g2);
        g3:v[3]$
        Dot_Product13:g1.A.transpose(g3);
        Dot_Product23:g2.A.transpose(g3);
(Dot_Product12) 0
(Dot_Product13) 0
(Dot_Product23) 0
(%i34)  U:matrix([g1[1],g2[1],g3[1]],[g1[2],g2[2],g3[2]],
                 [g1[3],g2[3],g3[3]])$
        _A_:transpose(U).A.U$ _B_:transpose(U).B.U$
        Matr_KF:[U,_A_, _B_];
(Matr_KF) [ | 1  -1/4  0 |  | 2  0    0  |  | 2  0    0  | ]
          | 0   1    1 |, | 0  3/8  0  |, | 0  3/8  0  |
          | 0   1/2  1 |  | 0  0   -1  |  | 0  0   -2  |
```

В результате выполнения программы по матрицам A, B данных квадратичных форм при помощи процедуры `eigenvectors(C)` вычисляются собственные значения и собственные векторы матрицы сопровождающего оператора (формируется соответствующий список). Для пары собственных векторов `v[1]`, `v[2]`, соответствующих одному собственному значению, проводится процесс ортогонализации относительно симметрической билинейной формы, в результате чего получается ортогональная относительно симметрической билинейной формы система векторов. Ответ выводится в виде списка матриц, в котором первая матрица есть матрица общего линейного преобразования, вторая и третья – матрицы канонического вида соответствующих квадратичных форм.

Рассмотрим третий случай, когда обе к.ф. $L_1\left(\bar{x}\right)=\bar{x}^T A\bar{x}$, $L_2\left(\bar{x}\right)=\bar{x}^T B\bar{x}$ являются вырожденными. Обе к.ф. порождают симметрические б.ф. $f\left(\bar{x},\bar{y}\right)=\bar{x}^T A\bar{y}$, $g\left(\bar{x},\bar{y}\right)=\bar{x}^T B\bar{y}$. Алгоритм построения общего канонического базиса для пары к.ф. основан на ряде понятий нуль-подпространства симметрической б.ф. как множества

$$N(f)=\left\{\bar{x}\in V:\ f\left(\bar{x},\ \bar{y}\right)=0,\ \forall\bar{y}\in V\right\}.$$

Это же множество будем называть нуль-подпространством к.ф. $L_1\left(\bar{x}\right)$, порожденной формой $f\left(\bar{x},\ \bar{y}\right)$, и обозначать $N\left(L_1\right)$.

Если к.ф. $L_1\left(\bar{x}\right)=\bar{x}^T A\bar{x}$ задана в базисе матрицей A, то множество $N\left(L_1\right)$ относительно этого базиса описывается однородной системой линейных

уравнений, основной матрицей которой является матрица A [нуль-подпространство $N(L_1)$ к.ф. совпадает с нуль-подпространством матрицы A].

Нуль-подпространством $N(L_1, L_2)$ пары (L_1, L_2) к.ф. $L_1(\bar{x})$, $L_2(\bar{x})$ называется множество: $N(L_1, L_2) = N(L_1) \cap N(L_2)$. При этом пара (L_1, L_2) называется *невырожденной*, если размерность $\dim N(L_1, L_2)$ подпространства $N(L_1, L_2)$ равна нулю. В противном случае пара (L_1, L_2) называется *вырожденной*.

Если пара (L_1, L_2) к.ф. является невырожденной, то её *собственным подпространством* называется нуль-подпространство к.ф. $\mu L_1(\bar{x}) - \lambda L_2(\bar{x})$, если:

1) его размерность больше нуля;
2) числа μ, λ не равны нулю одновременно.

При этом пару (μ, λ) называют *собственной парой чисел* пары (L_1, L_2). Собственное подпространство пары (L_1, L_2) будем обозначать в виде

$$L(\mu, \lambda) = N(\mu L_1 - \lambda L_2) = \left\{\bar{x} \in V : \mu L_1(\bar{x}) - \lambda L_2(\bar{x}) = 0\right\},$$
$$\dim L(\mu, \lambda) > 0, \ \mu^2 + \lambda^2 \neq 0.$$

У любой пары (L_1, L_2) вырожденных квадратичных форм $L_1(\bar{x})$, $L_2(\bar{x})$ обязательно существуют по крайней мере два собственных подпространства:
1) при $\mu = 1, \lambda = 0$ собственное подпространство $L(1, 0) = N(L_1)$;
2) при $\mu = 0, \lambda = 1$ собственное подпространство $L(0, 1) = N(L_2)$.

Предположим, что необходимо найти собственное подпространство $L(\mu, \lambda)$ пары (L_1, L_2) к.ф. при условии, что $\mu \neq 0, \lambda \neq 0$. В этом случае необходимо выполнение условия $\det\left(A - \frac{\lambda}{\mu} B\right) = 0$. При этом пара (L_1, L_2) к.ф. имеет собственное подпространство $L(1, \lambda)$ $(\lambda \neq 0)$ тогда и только тогда, когда число λ удовлетворяет характеристическому уравнению пары (L_1, L_2):

$$\det(A - \lambda B) = 0. \qquad (5.16)$$

Множество Λ корней характеристического уравнения (5.16) называется спектром пары (L_1, L_2).

Перечислим основные свойства собственных подпространств пары (L_1, L_2) к.ф.:

1) различные собственные подпространства невырожденной пары (L_1, L_2) к.ф. пересекаются только по нулевому вектору;

2) два различных собственных подпространства невырожденной пары (L_1, L_2) квадратичных форм ортогональны (сопряжены) относительно

симметричных билинейных форм, порождающих эти квадратичные формы: при всех $\bar{x} \in L(1,\ \lambda_i)$, $\bar{y} \in L(1,\ \lambda_j)$ ($\lambda_i \neq \lambda_j$ при $i \neq j$)

$$f(\bar{x},\ \bar{y}) = 0,\ g(\bar{x},\ \bar{y}) = 0; \qquad (5.17)$$

3) если характеристическое уравнение (5.16) невырожденной пары $(L_1,\ L_2)$ к.ф. имеет степень выше нулевой, то собственные подпространства этой пары образуют прямую сумму подпространств.

Теорема 2 (необходимое и достаточное условия существования общего диагонализирующего базиса для пары к.ф.) Невырожденная пара $(L_1,\ L_2)$ к.ф. в пространстве V имеет общий диагонализирующий базис тогда и только тогда, когда сумма размерностей всех собственных подпространств этой пары равна размерности всего пространства.

Рассмотрим *доказательство* теоремы.

□ Необходимость. Пусть $\dim V = n$ и для невырожденной пары $(L_1,\ L_2)$ существует общий диагонализирующий базис $\mathbf{B}_N = (\bar{e}_1,\ \bar{e}_2,\ ...,\ \bar{e}_n)$. В этом базисе матрицы $A',\ B'$ к.ф. $L_1(\bar{x}),\ L_2(\bar{x})$ имеют диагональный вид:

$$A' = diag(a'_1,\ a'_2,\ ...,\ a'_n),\ B' = diag(b'_1,\ b'_2,\ ...,\ b'_n).$$

Предположим, что $rang B' = r$, причем $r < n$. Тогда без ограничения общности можно считать, что в силу закона инерции к.ф. в последовательности канонических коэффициентов $b'_1,\ b'_2,\ ...,\ b'_n$ первые $n-r$ коэффициентов равны нулю:

$$\underbrace{0,\ ...,\ 0}_{n-r},\ b'_{n-r+1},\ ...,\ b'_n.$$

При этом, так как пара $(L_1,\ L_2)$ невырожденная, то в последовательности канонических коэффициентов $a'_1,\ a'_2,\ ...,\ a'_n$ первые $n-r$ коэффициентов обязательно отличны от нуля.

Так как $rang B' = r$ ($r < n$), то собственное подпространство $L(0,\ 1)$ имеет размерность $n-r$. Остальные собственные подпространства имеют вид:

$L(1,\ \lambda)$, где $\lambda \in \Lambda$ – корень уравнения (5.16), которое в нашем случае имеет вид

$$\det(A' - \lambda B') = \underbrace{a'_1 \cdot a'_2 \cdot a'_{n-r}}_{n-r} \cdot \underbrace{(a'_{n-r+1} - \lambda b'_{n-r+1}) \cdot ... \cdot (a'_n - \lambda b'_n)}_{r} = 0.$$

Размерность каждого собственного подпространства $L(1,\ \lambda)$ в этом случае совпадает с алгебраической кратностью корня λ этого уравнения. Следовательно, сумма размерностей всех собственных подпространств $L(1,\ \lambda)$ ($\lambda \in \Lambda$) равна r. В результате

$$\dim L(0,\ 1) + \sum_{\lambda \in \Lambda} \dim L(1,\ \lambda) = (n-r) + r = n = \dim V.$$

Достаточность. Пусть $\dim L(0,\ 1)+\sum\limits_{\lambda\in\Lambda}\dim L(1,\ \lambda)=n.$ В этом случае степень характеристического уравнения (5.16) пары $(L_1,\ L_2)$ квадратичных форм выше нулевой. Тогда собственные подпространства пары образуют прямую сумму подпространств. Значит, объединение базисов всех собственных подпространств будет являться базисом всего пространства V.

Покажем, как построить общий диагонализирующий базис. Здесь возможны два случая.

Случай 1. Пусть размерность каждого собственного подпространства $L(1,\ \lambda)$ $(\lambda\in\Lambda)$, $L(0,\ 1)$ равна 1. В этом случае общий диагонализирующий базис $\mathbf{B}_N=(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_n})$ представляет собой объединение базисов всех собственных подпространств. Для любых векторов $\overline{e_i},\ \overline{e_j}\in\mathbf{B}_N$ $(i\neq j;\ i,\ j=\overline{1,\ n})$, принадлежащих различным собственным подпространствам, имеем ортогональность относительно симметричных билинейных форм:

$$f(\overline{e_i},\ \overline{e_j})=\overline{e_i}^T A\overline{e_j}=0,\ g(\overline{e_i},\ \overline{e_j})=\overline{e_i}^T B\overline{e_j}=0\ \ (i\neq j;\ i,\ j=\overline{1,\ n}).\quad (5.18)$$

Если при этом составить матрицу $U=(\overline{e_1}|\overline{e_2}|...|\overline{e_n})$, столбцами которой являются векторы базиса $\mathbf{B}_N=(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_n})$, то в силу выполнения равенств (5.18) в этом базисе матрицы $A',\ B'$ к.ф. $L_1(\overline{x}),\ L_2(\overline{x})$ примут диагональный вид:

$$A'=U^TAU=diag\{a'_1,\ a'_2,\ ...,\ a'_n\},\ B'=U^TBU=diag\{b'_1,\ b'_2,\ ...,\ b'_n\},$$

где $a'_i=\overline{e_i}^T A\overline{e_i},\ b'_i=\overline{e_i}^T B\overline{e_i}\ (i=\overline{1,n})$.

Случай 2. Пусть размерность одного из собственных подпространств пары форм больше единицы. Пусть $\dim L(1,\ 0)=k>1$ и $\mathbf{G}=(\overline{g_1},\ \overline{g_2},\ ...,\ \overline{g_k})$ есть базис $L(1,\ 0)$. При этом

$$g(\overline{g_i},\ \overline{g_j})=\overline{g_i}^T B\overline{g_j}\neq 0\ (i\neq j;\ i,\ j=\overline{1,k}),$$

то есть векторы, взятые из собственного подпространства $L(1,\ 0)$, не ортогональны относительно симметрической билинейной формы $g(\overline{x},\ \overline{y})$ [при этом обязательно $f(\overline{g_i},\ \overline{g_j})=0\ (i\neq j;\ i,\ j=\overline{1,k})$].

Проведя для базиса $\mathbf{G}=(\overline{g_1},\ \overline{g_2},\ ...,\ \overline{g_k})$ процесс ортогонализации относительно симметрической б.ф. $g(\overline{x},\ \overline{y})$ (см. случай 2), получим новый базис $\mathbf{G}_0=(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_k})$, ортогональный относительно обеих б.ф.:

$f\left(\overline{e_i},\ \overline{e_j}\right)=0,\ g\left(\overline{e_i},\ \overline{e_j}\right)=0\ \left(i\neq j;\ i,j=\overline{1,k}\right).$

Пусть $\dim L\left(1,\ \lambda\right)=k>1\quad(\lambda\neq 0)$ и $\mathbf{G}=\left(\overline{g_1},\ \overline{g_2},\ ...,\ \overline{g_k}\right)$ есть базис подпространства $L\left(1,\ \lambda\right)$. В этом случае для любого вектора $\overline{z}\in V$:

$$f\left(\overline{g_i},\ \overline{z}\right)-\lambda g\left(\overline{g_i},\ \overline{z}\right)=0,\ \left(i=\overline{1,k}\right).$$

Ортогонализировав базис $\mathbf{G}$ относительно симметрической б.ф. $g\left(\overline{x},\ \overline{y}\right)$, получим новый базис $\mathbf{G}_0=\left(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_k}\right)$ такой, что при всех $\overline{z}\in V$

$$\begin{cases} f\left(\overline{e_i},\ \overline{e_j}\right)=0\ \left(i\neq j,\ i,j=\overline{1,k}\right), \\ f\left(\overline{e_i},\ \overline{z}\right)-\lambda g\left(\overline{e_i},\ \overline{z}\right)=0\ \left(i=\overline{1,k}\right). \end{cases}$$

Положив в последней системе $\overline{z}=\overline{e_j}\in L\left(1,\ \lambda\right)\ \left(i\neq j,\ \lambda\neq 0\right)$, получим

$$\begin{cases} f\left(\overline{e_i},\ \overline{e_j}\right)=0, \\ f\left(\overline{e_i},\ \overline{e_j}\right)-\lambda g\left(\overline{e_i},\ \overline{e_j}\right)=0\ \left(i\neq j,\ i,j=\overline{1,k}\right), \end{cases}$$

что возможно только в том случае, когда

$$f\left(\overline{e_i},\ \overline{e_j}\right)=0,\ g\left(\overline{e_i},\ \overline{e_j}\right)=0\ \left(i\neq j,\ i,j=\overline{1,k}\right).$$

Итак, построенный базис $\mathbf{G}_0=\left(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_k}\right)$ является ортогональным относительно симметрических б.ф. $f\left(\overline{x},\ \overline{y}\right),\ g\left(\overline{x},\ \overline{y}\right)$.

Если $\dim L\left(0,\ 1\right)=k>1,$ то базис в $L\left(0,\ 1\right)$ необходимо ортогонализировать относительно симметрической б.ф. $f\left(\overline{x},\ \overline{y}\right)$.

В результате общий диагонализирующий базис $\mathbf{B}_N$ получится путем объединения построенных базисов каждого собственного подпространства. Теорема доказана.

На основании теоремы 2 можно составить алгоритм построения общего диагонализирующего базиса невырожденной пары $\left(L_1,\ L_2\right)$ вырожденных к.ф. $L_1\left(\overline{x}\right)=\overline{x}^T A\overline{x},\ L_2\left(\overline{x}\right)=\overline{x}^T B\overline{x}$:

1) найти собственные подпространства $L\left(1,\ 0\right)$ и $L\left(0,\ 1\right)$ пары $\left(L_1,\ L_2\right)$, базисы в этих подпространствах и их размерности. Проверить, является ли пара $\left(L_1,\ L_2\right)$ квадратичных форм невырожденной, найдя их пересечение;

2) составить характеристическое уравнение ненулевой степени, найти спектр Λ пары форм;

3) для каждого собственного числа $\lambda_1,\ ...,\ \lambda_m\in\Lambda$ найти соответствующее собственное подпространство $L\left(1,\ \lambda_i\right)$, его базис и размерность;

4) найти сумму размерностей всех полученных собственных подпространств: $\dim L(0,\ 1)+\dim L(1,\ 0)+\sum\limits_{\substack{\lambda_i\in\Lambda,\\ \lambda_i\neq 0}}\dim L(1,\ \lambda_i)$. Если полученная сумма равна размерности всего пространства, то по теореме 2 пара $(L_1,\ L_2)$ к.ф. имеет общий диагонализирующий базис;

5) построить общий диагонализирующий базис $\mathbf{B}_N=(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_n})$ для пары к.ф. При этом, если размерность каждого собственного подпространства равна единице [спектр Λ пары $(L_1,\ L_2)$ простой], то искомый диагонализирующий базис $\mathbf{B}_N$ получается объединением полученных базисов (по одному базисному вектору) каждого собственного подпространства. Если размерность собственного подпространства больше единицы [спектр Λ пары $(L_1,\ L_2)$ сложный], то предварительно провести процесс ортогонализации относительно соответствующей симметрической б.ф.

6) составить матрицу $U=(\overline{e_1}|\overline{e_2}|...|\overline{e_n})$, столбцами которой являются векторы диагонализирующего базиса $\mathbf{B}_N=(\overline{e_1},\ \overline{e_2},\ ...,\ \overline{e_n})$. В результате неособенного линейного преобразования $\overline{x}=U\overline{y}$ в базисе $\mathbf{B}_N$ матрицы к.ф. примут диагональный вид:

$$A'=U^TAU=diag\{a'_1,\ a'_2,\ ...,\ a'_n\},\ B'=U^TBU=diag\{b'_1,\ b'_2,\ ...,\ b'_n\},$$

где $a'_i=\overline{e_i}^T A\overline{e_i},\ b'_i=\overline{e_i}^T B\overline{e_i}\ (i=\overline{1,n})$.

Ниже представлен фрагмент программы в СКА Maple приведения пары к.ф. к каноническому виду одним линейным преобразованием (случай вырожденности форм).

```
> A:=Matrix(n,n, a): B:=Matrix(n,n, b):
  L[1]:=simplify(Transpose(X).A.X):
  L[2]:=simplify(Transpose(X).B.X): [A,B];
```

$$\left[\begin{bmatrix} 6 & -12 & 0 & 4\\ -12 & 24 & 0 & -8\\ 0 & 0 & 0 & 0\\ 4 & -8 & 0 & 3\end{bmatrix},\begin{bmatrix} 5 & -12 & 0 & 3\\ -12 & 21 & -4 & -7\\ 0 & -4 & -2 & 0\\ 3 & -7 & 0 & 2\end{bmatrix}\right]$$

```
> r[1]:=Rank(A): NullSpace_A:=NullSpace(A);
  Dim_NullSpace_A:=n-r[1];
  for i from 1 by 1 to Dim_NullSpace_A do
    e[i]:=NullSpace_A[i]
  end do:
```

$$NullSpace\_A := \left\{\begin{bmatrix} 0\\ 0\\ 1\\ 0\end{bmatrix},\begin{bmatrix} 2\\ 1\\ 0\\ 0\end{bmatrix}\right\}$$

$$Dim\_NullSpace\_A := 2$$

```
> # проводим процесс ортогонализации для векторов e[1], e[2]
относительно СБФ B(x,y)
> g[1]:=e[1]:
 g[2]:=e[2]-Transpose(g[1]).B.e[2])/(Transpose(g[1]).B.g[1])*g[1]:
> e[1]:=g[1]: e[2]:=g[2]: NullSpace_A:={e[1], e[2]};
```

$$NullSpace\_A := \left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ -2 \\ 0 \end{bmatrix} \right\}$$

```
> r[2]:=Rank(B):
 NullSpace_B:=NullSpace(B);Dim_NullSpace_B:=n-r[2];
```

$$NullSpace\_B := \left\{ \begin{bmatrix} -3 \\ -1 \\ 2 \\ 1 \end{bmatrix} \right\}$$

$$Dim\_NullSpace\_B := 1$$

```
> IntersectionBasis([NullSpace_A, NullSpace_B]);
```

$$\{\ \}$$

```
> C:=A-lambda*B: P:=Determinant(C); Spektr:=[solve(P,lambda)];
 j:=1:
 for i from 1 by 1 to 3 do
  if Spektr[i]<>0 then lambda[j]:=Spektr[i]: j:=j+1:
 end if:
 end do:
```

$$P := -4\,\lambda^2 + 2\,\lambda^3$$

$$Spektr := [0, 0, 2]$$

```
> lambda:=lambda[1]: r[3]:=Rank(C): NullSpace_C:=NullSpace(C);
 Dim_NullSpace_C:=n-r[3];
```

$$NullSpace\_C := \left\{ \begin{bmatrix} -\frac{1}{2} \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

$$Dim\_NullSpace\_C := 1$$

```
> Summa:=SumBasis([NullSpace_A, NullSpace_B, NullSpace_C]);
 U:=Summa[1]:
 for i from 2 by 1 to n do
  U:=<U|Summa[i]>;
 end do:
 U:=U;
```

$$U := \begin{bmatrix} -\frac{1}{2} & 0 & 2 & -3 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & -2 & 2 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

```
>_A_:=simplify(Transpose(U).A.U);_B_:=simplify(Transpose(U).B.U);
 K[1]:=simplify(Transpose(Y)._A_.Y)[1,1];
 K[2]:=simplify(Transpose(Y)._B_.Y)[1,1];
```

$$\_A\_ := \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \_B\_ := \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$K_1 := \frac{1}{2} y_1^{\,2} + y_4^{\,2} \quad K_2 := \frac{1}{4} y_1^{\,2} - 2\, y_2^{\,2} + y_3^{\,2}$$

В программе при помощи процедуры `NullSpace` находятся базисы (переменные `NullSpace_A, NullSpace_B`) и размерности (переменные `Dim_NullSpace_A, Dim_NullSpace_B`) собственных подпространств $L(1,\ 0)$ и $L(0,\ 1)$ пары $(L_1,\ L_2)$. Для базиса `NullSpace_A` проводится процесс ортогонализации относительно симметрической б.ф. $g(\bar{x},\ \bar{y})$. С помощью процедуры `IntersectionBasis` проверяется, что собственные подпространства `NullSpace_A, NullSpace_B` не пересекаются. При помощи процедуры `solve(P, lambda)` определяется спектр (переменная `Spektr`) пары форм. Переменная `NullSpace_C` хранит базис собственного подпространства $L(1,\ \lambda_i)$ для $\lambda_i \neq 0$. Процедура `SumBasis([NullSpace_A,NullSpace_B,NullSpace_C])` формирует общий диагонализирующий базис $\mathbf{B}_N$ для пары к.ф. (соответственно матрица `U` есть матрица линейного преобразования). В итоге матрицы к.ф. принимают в базисе $\mathbf{B}_N$ диагональный вид (матрицы `_A_, _B_`).

Ниже представлена реализация алгоритма приведения пары квадратичных форм к каноническому виду в СКА wxMaxima.

```
A:matrix([6,-12,0,4],[-12,24,0,-8],[0,0,0,0],[4,-8,0,3])$
B:matrix([5,-12,0,3],[-12,21,-4,-7],[0,-4,-2,0],[3,-7,0,2])$
```

```
Dim_A:n-rank(A);
NullSpace_A:nullspace(A);
```

```
(Dim_A) 2
0 errors, 0 warnings
```

(NullSpace_A) $\mathrm{span}\left(\begin{bmatrix} 0 \\ 0 \\ 4 \\ 0 \end{bmatrix}, \begin{bmatrix} 4 \\ 2 \\ 0 \\ 0 \end{bmatrix}\right)$

```
(%i10) v1:transpose(matrix([0],[0],[4],[0]))$
       v2:transpose(matrix([4],[2],[0],[0]))$
       e1:v1$ e2:v2-(e1.B.v2/e1.B.e1)*e1$
       e1.B.transpose(e2);
(%o10) 0
```

```
(%i12)  Dim_B:n-rank(B);
        NullSpace_B:nullspace(B);
(Dim_B) 1
```

$$(\text{NullSpace\_B})\quad \text{span}\left(\begin{bmatrix}-6\\-2\\4\\2\end{bmatrix}\right)$$

```
(%i15)  e3:transpose(matrix([-6],[-2],[4],[2]))$
        e1.A.transpose(e3); e2.A.transpose(e3);
(%o14)  0
(%o15)  0
(%i19)  C:A-k*B$
        P:determinant(C)$ P:radcan(P);
        Spectr:solve(P,k);
(P)     2 k^3-4 k^2
(Spectr) [k=0,k=2]
(%i23)  k:2$ C:A-k*B$
        Dim_C:rank(C)$
        NullSpace_C:nullspace(C);
```

$$(\text{NullSpace\_C})\quad \text{span}\left(\begin{bmatrix}-16\\0\\0\\32\end{bmatrix}\right)$$

```
(%i27)  e4:transpose(matrix([-16],[0],[0],[32]))$
        e1.A.transpose(e4); e2.A.transpose(e4); e3.A.transpose(e4);
(%o25)  0
(%o26)  0
(%o27)  0
(%i31)  U:matrix([e1[1][1],e2[1][1],e3[1][1],e4[1][1]],
                 [e1[1][2],e2[1][2],e3[1][2],e4[1][2]],
                 [e1[1][3],e2[1][3],e3[1][3],e4[1][3]],
                 [e1[1][4],e2[1][4],e3[1][4],e4[1][4]])$
        _A_:transpose(U).A.U$ _B_:transpose(U).B.U$
        Matr_KF:[U,_A_, _B_];
```

$$(\text{Matr\_KF})\quad \left[\begin{bmatrix}0&4&-6&-16\\0&2&-2&0\\4&-4&4&0\\0&0&2&32\end{bmatrix},\begin{bmatrix}0&0&0&0\\0&0&0&0\\0&0&4&0\\0&0&0&512\end{bmatrix},\begin{bmatrix}-32&0&0&0\\0&4&0&0\\0&0&0&0\\0&0&0&256\end{bmatrix}\right]$$

В результате выполнения программы по матрицам A, B данных квадратичных форм при помощи процедур `nullspace(A)` и `nullspace(B)` находятся базисы соответствующих нуль-подпространств $L(1, 0)$ и $L(0, 1)$

[для базиса в подпространстве $L(1, 0)$ проводится процесс ортогонализации относительно симметрической билинейной формы $g(\bar{x}, \bar{y})$]. По спектру $\Lambda = \{0, 2\}$ пары форм для числа 2 определяется базис подпространства $L(1, 2)$. Матрица U преобразования, приводящего обе квадратичные формы к каноническому виду, формируется из векторов базисов подпространств $L(1, 0)$, $L(0, 1)$, $L(1, 2)$.

Литература к главе 5

1. Аладьев В.З., Бойко В.К., Ровба Е.А. Программирование в пакетах Maple и Mathematica: Сравнительный аспект: монография, 2011. 517 с.

2. Беллман Р. Введение в теорию матриц. М.: Наука, 1969.

3. Бортаковский А.С. Линейная алгебра в примерах и задачах: учеб. пособие/ А.С. Бортаковский, А.В. Пантелеев. – М.: Высш. шк., 2005, 591 с.

4. Гантмахер Ф.Р. Теория матриц. М.: Наука, 1988.

5. Ильин В.А., Позняк Э.Г. Линейная алгебра. М.: Наука, 1999. 304 с.

6. Прасолов В.В. Задачи и теоремы линейной алгебры. М.: Наука, 1996.

7. Проскуряков И. В. Сборник задач по линейной алгебре. 5-е изд. М.: Наука, 1974. 384 с.

8. Стренг Г. Линейная алгебра и ее применения / пер. с англ. под ред. Г.И. Марчука. М.: Мир, 1980. 456 с.

9. Фаддеев Д.К., Фаддеева В.Н. Вычислительные методы линейной алгебры. М.: Физматгиз, 1963.

10. Халмош П. Конечномерные векторные пространства. М.: Мир, 1970.

11. Хорн Р., Джонсон Ч. Матричный анализ. М.: Мир, 1989.

12. Яцкин Н.И. Линейная алгебра: теоремы и алгоритмы: учеб. пособие / Н.И. Яцкин. Иваново: Иван. гос. ун-т, 2008. 607 с.

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ
В ПРОЦЕССЕ ОБУЧЕНИЯ

Коллективная монография

Под редакцией
доктора физико-математических наук, профессора,
директора лаборатории системного анализа
Миронова Валентина Васильевича

Подписано в печать _____. Формат бумаги _____.
Бумага _____. Печать _____. Усл. печ. л. _____.
Тираж 50 экз.
Типография Book Jet.
390046, г. Рязань, Скорбящинский проезд, д. 20, оф. 23.