В.Б. Маничев

МЕТОДЫ ЧИСЛЕННОГО РЕШЕНИЯ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ, НЕЛИНЕЙНЫХ И ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ С ГАРАНТИРОВАННОЙ ДОСТОВЕРНОСТЬЮ И ТОЧНОСТЬЮ

Лекции по курсам

“Основы автоматизированного проектирования,

«Численные методы»

для специальности

2301040065 «Системы автоматизированного проектирования»

кафедра РК6 (САПР) МГТУ имени Н.Э.Баумана

МОСКВА 20..

Оглавление

ПРЕДИСЛОВИЕ 3

1. Достоверность и точность математического моделирования, неэквивалентность некоторых матрично-векторных преобразований на компьютере, трудности программирования математических вычислений с вещественными числами с плавающей точкой 8

1.1. Источники ошибок в процессе математического моделирования 8

1.2. Представление чисел при компьютерных вычислениях, вещественные числа с плавающей точкой и целые числа 12

1.3. Форматы представления вещественных чисел с плавающей точкой..... 14

1.4. Диапазон и точность представления вещественных чисел в форматах IEEE754. 17

1.5. Проблемы компьютерных вычислений, вызванные использованием стандарта IEEE754. 21

1.5.1. Ошибки связанные с точностью представления вещественных чисел в формате IEEE754. 22

1.5.2. Ошибки связанные с неправильным приведением типов данных. 22

1.5.3. Ошибки вызванные сдвигом мантисс. 22

1.5.4. Ошибки вызванные округлением. 23

1.5.5. Ошибки на границе норма/денорма числа. 24

2. Численные методы решения систем обыкновенных дифференциальных уравнений 26

2.1 Разновидности обыкновенных дифференциальных уравнений 26

2.2. Общий алгоритм численного решения систем ОДУ-ДАУ..... 31

3. Численные методы решения систем нелинейных алгебраических уравнений 32

4. Численные методы решения систем линейных алгебраических уравнений 32

5. Цикл лабораторных работ..... 32

ЛИТЕРАТУРА. 32

ПРЕДИСЛОВИЕ

В настоящее время (2012 г.) можно выделить два основных направления использования систем автоматизированного, компьютерного проектирования (САПР) разнообразных изделий и объектов:

1. Промышленные САПР:

- САПР (CAD/CAM/CAE/PDM системы) для проектирования изделий машиностроения (программные продукты компаний Dassault Systemes (CATIA / DELMIA / SIMULIA / ENOVIA…), Siemens PLM Software (UNIGRAPHICS(NX)-CAD / NX-CAM / NX CAE / Teamcenter…), PTC (CAD/CAM/ система Creo / MathCAD / WindChill…), Autodesk (Inventor / CAM/CAE системы других компаний / PDM системы Autodesk…), АСКОН (КОМПАС / CAM системы других компаний / CAE системы АСКОН / ЛОЦМАН:PLM…), и др.);

- САПР для проектирования изделий электроники и электротехники (программные продукты компаний Cadance (Virtuoso, Allegro, OrCAD…), Synopsys (SynplifyPro, ModelSim…), Mentor Graphics (Calibre nmDRC, Nucleus RTOS, FloEFD…), PCSCHEMATIC (Automation…) и др.);

- САПР для проектирования строительных и архитектурных объектов (программные продукты компаний Autodesk (Autodesk Revit…), Bentley Systems (Bentley Architectural…) и др.);

- САПР для проектирования промышленных предприятий и проектирования инфраструктуры в течение ее жизненного цикла, т.е. при проектировании, создании и эксплуатации зданий, мостов, транспортных сетей, предприятий водо- тепло- энерго- снабжения, очистки воды и т.п, а также геоинформационные системы, которые являются неотъемлемой частью данных САПР (программные продукты компаний Autodesk (Autodesk AutoCAD Civil 3D, Autodesk Map 3D…), Bentley Systems (Bentley AutoPlant, Bentley PowerMap…) и др.).

2. Непромышленные САПР:

- САПР в медицине и биологии, например, для проектирования протезов, медицинского оборудования, для проектирования хирургических операций - CAS (Computer Aided Surgery) системы, программы САПР в генной инженерии и др.

- САПР в спорте, например, для проектирования спортивного инвентаря, спортивных тренажеров и др.

Автоматизация конструкторского проектирования (CAD, CAE, PDM подсистемы САПР) промышленных изделий и объектов включает в себя три аспекта:

1. Автоматизация инженерного конструирования промышленных изделий и объектов с помощью соответствующих параметрических 3D и 2D CAD (Computer Aided Design) подсистем САПР.

2. Компьютерные инженерные расчеты, математическое моделирование, анализ и оптимизация промышленных изделий и объектов с помощью CAE (Computer Aided Engineering) подсистем САПР.

3. Управление инженерным документооборотом, автоматизация коллективной работы инженеров-проектировщиков с помощью PDM (Product Data Management) подсистем САПР. Подсистемы PDM САПР составляют основу управления всем жизненным циклом проектируемых промышленных изделий и объектов: CALS (Continuous Acquisition and Life cycle Support) и PLM (Product Lifecycle Management) технологий. CALS технологии это непрерывная информационная поддержка поставок и жизненного цикла промышленных изделий и объектов) — современный подход к проектированию и производству высокотехнологичной и наукоёмкой продукции, заключающийся в использовании компьютерной техники и современных информационных технологий на всех стадиях жизненного цикла промышленных изделий и объектов. За счет непрерывной информационной поддержки обеспечиваются единообразные способы управления процессами и взаимодействия всех участников этого цикла: заказчиков продукции, поставщиков/производителей продукции, эксплуатационного и ремонтного персонала. Информационная поддержка реализуется в соответствии с требованиями системы международных стандартов (CALS стандартов), регламентирующих правила указанного взаимодействия преимущественно посредством электронного обмена данными. PLM технологии это обеспечение взаимодействия как средств автоматизации разных производителей, так и различных автоматизированных систем многих предприятий, то есть технологии PLM являются основой, интегрирующей информационное пространство, в котором функционируют вышеуказанные подсистемы САПР, системы ERP (Enterprise Resource Planning, - планирование финансовых ресурсов предприятий) и другие автоматизированные системы промышленных предприятий.

CAE технологии и подсистемы САПР становятся всё более востребованными при разработке технических изделий и объектов. Какие бы не применялись подходы, становится очевидным, что математическое моделирование процессов и инженерный анализ (Simulation & Analysis - S&A), которые пока являются неотъемлемой частью цикла проектирования в основном в высокотехнологичных и наукоёмких отраслях, постепенно становятся таковыми и в отраслях, занятых выпуском продукции массового спроса. Поэтому сегодняшнему среднестатистическому инженеру уже бывает недостаточно только встроенных в САПР средств инженерного анализа (embedded CAE), а поставщики САПР, которые ранее не обращали серьёзного внимания на CAE технологии, уже проявили свое стремление расширить функ-

ционал поставляемых САПР за счёт специализированных CAE подсистем. Кроме того, миниатюризация электроники стимулирует объединение механических и электронных компонентов в одном конструктиве, - развивается мехатроника (Mechatronics: Mechanical and Electronics Engineering) - новая научно-техническая дисциплина, которая изучает построение электромеханических систем нового поколения, основанных на применении знаний в области механики, электроники, микропроцессорной техники, информатики и компьютерного управления движением машин и агрегатов. Создание и развитие электромеханических изделий приводит к необходимости интеграции подсистем механического, электротехнического и электронного проектирования, а значит и к интеграции соответствующих CAE технологий для математического моделирования и инженерного анализа, т.е. интеграцию CAE подсистем для машиностроения - то есть MCAE подсистем (Mechanical Computer Aided Engineering) с системами электронного проектирования (Electronic Design Automation – EDA) или с подсистемами электротехнического и электронного проектирования ECAE (Electrical or Electronic CAE) подсистемами.

CAE подсистемы САПР в большинстве случаев основаны на численном решении различных математических уравнений. Например, при математическом моделировании и инженерном анализе разнообразных технических систем и объектов на основе дифференциальных уравнений можно выделить два основных направления:

1. Моделирование в пространственно-временной области на основе дифференциальных уравнений в частных производных (ДУЧП – PDE (Partial Differential Equations)), программные комплексы: ANSYS (Ansys), MSC Nastran (MSC Software), NX Nastran (Siemens PLM Software) Simulia Abaqus (Dassault Systemes) и др.

2. Моделирование во временной области на основе обыкновенных дифференциальных уравнений (ОДУ - ODE (Ordinary Differential Equations)), как разрешенных относительно производных, так и систем дифференциально-алгебраических уравнений (ДАУ - DAE (Differential Algebraic Equations)), не разрешенных относительно производных, программные комплексы: SPICE (Cadance, Synopsys, Mentor Graphics), MSC Adams, MSC Easy5 (MSC Software, Boeing), EULER (AutoMechanics), MATLAB-Simulink (MathSoft), Maple-MapleSim (Maple), Mathcad (PTC), МВТУ (МГТУ имени Н.Э. Баумана), ПА9 (МГТУ имени Н.Э. Баумана) и др.

Численное решение дифференциальных уравнений, в свою очередь во многих численных методах их решения, сводится к численному решению соответствующих систем линейных алгебраических уравнений (СЛАУ). Высокий порядок таких систем и необходимость их многократного решения в процессе проектирования предъявляют жесткие требования к эффективности применяемых численных методов. Эффективность численных методов решения систем ДУЧП, ОДУ-ДАУ и СЛАУ включает в себя два аспекта:

1. Достоверность и точность получаемых решений данных уравнений.

2. Затраты машинного времени и оперативной памяти компьютера на получение решений.

Фантастический рост производительности вычислительной техники в последнее время выдвигает на первый план проблему получения достоверных и точных решений. В CAE подсистемах должна быть обеспечена 100 процентная гарантия получения достоверности и требуемой точности, выдаваемых инженерам-проектировщикам результатов компьютерных расчетов и математического моделирования, т.к. ошибки в принятии проектных решений для промышленных изделий и объектов, основанном на этих результатах, стоят очень дорого. Ошибки и неточности, недавно обнаруженные в традиционных методах и алгоритмах расчета и в популярных математических пакетах (MATLAB, Maple, Mathcad, Mathematica др.) могут быть причиной многих аварий и катастроф, если они попадут в конечную конструкторскую документацию [1].

Президент совета по конкурентоспособности США заявлял: «Технологии, таланты и деньги доступны многим странам. Поэтому США стоят перед лицом непредсказуемых зарубежных экономических конкурентов. Страна, желающая победить в конкуренции, должна победить в вычислениях». Но вычисления, как указано выше, включают в себя 2 аспекта:

1. Производительность вычислений ("мускулы").

2. Достоверность и точность вычислений ("мозги").

Вложения в вычисления коммерчески не выгодны, поэтому, как правило, это исключительно бюджетные средства. В США в 2005-2008 годах, например, тратили от 2 до 6 млрд. долл. ежегодно на поддержку высокопроизводительных вычислений, поэтому по 1-му аспекту ("мускулы") США в 2009-м году были безусловным лидером с суммарной производительностью суперкомпьютеров в 16 416 TFPOPS (на втором месте была объединенная Европа – 7667 TFPOPS, на третьем месте Китай – 2993 TFPOPS). В России было 646 TFPOPS, но сейчас вкладывается достаточно средств, чтобы выправить ситуацию. Огромные средства вкладывает в высокопроизводительные вычисления Япония (в 2012 году самый мощный суперкомпьютер в мире, - K-computer с производительностью 10,4 PFLOPS, был разработан именно в Японии).

В то же время профессор СПбГУ Петров Юрий Петрович в своих книгах [1,2] показал, что для вычислений, связанных с проектированием и созданием сложной и опасной техники (аэрокосмическая техника, атомные станции и т.п.), 2-ой аспект ("мозги") намного важнее 1-го. Россия пока опережает по 2-му аспекту лучшие зарубежные разработки (достаточно назвать разработки математического и программного обеспечения для компьютеров и суперкомпьютеров ИВМ РАН, ИММ РАН, ППМ РАН, ИППМ РАН, ВЦ РАН и других институтов РАН, МГУ имени М.В. Ломоносова, МГТУ имени Н.Э. Баумана, СПбГУ, ТГУ и других универси-

тетов), так как даже разработки кафедры РК6 (САПР) МГТУ имени Н.Э. Баумана по достоверности и точности решений систем ОДУ-ДАУ и СЛАУ значительно превзошли (библиотека программ SADEL) лучшие зарубежные программные математические библиотеки и системы: LINPACK, LAPACK, NAG (Кембридж), Intel MKL, IMSL, MAGMA , MATLAB, Maple, Mathcad, Mathematica, что показано в [4].

При разработке программ математического моделирования технических систем и объектов необходимо учесть следующее:

1. Большинство инженеров-проектировщиков не являются специалистами в численных методах и программах для решения систем ОДУ-ДАУ и СЛАУ, поэтому достоверность и требуемая точность должна быть обеспечена для параметров программ-решателей этих уравнений, рекомендуемых для этих решателей по умолчанию.

2. При базовом, начальном математическом моделировании реальных технических систем и объектов не требуется высокая математическая точность выдаваемых пользователю результатов, т.к. параметры математических моделей этих изделий и объектов получены, как правило, экспериментально с невысокой математической точностью. Поэтому требуемая математическая точность конечных результатов моделирования по умолчанию может быть невысокой, но достоверность результатов и требуемая точность должны быть обеспечены со 100 процентной гарантией.

Как будет показано ниже эти требования в известных математических пакетах программ для решения систем ОДУ-ДАУ и СЛАУ не решены, поэтому при разработке CAE подсистем САПР следует реализовывать соответствующие численные методы и алгоритмы с учетом этих требований.

В данном учебном пособии рассматриваются классические численные методы и алгоритмы для решения систем обыкновенных дифференциальных уравнений, нелинейных и линейных алгебраических уравнений и методы обеспечения гарантированной достоверности и требуемой точности получаемых при этом результатов. Кроме того излагаются кардинально новые идеи, которые еще не отражены в учебниках по вычислительной математике: отказ от приведения систем ОДУ к нормальной форме Коши, разрешенной относительно производных, а также отказ от каких-либо численных эквивалентных преобразований исходных уравнений математических моделей и исходных данных.

Рассматривается также цикл лабораторных работ по решению соответствующих уравнений на языке программирования Си в стандартах IEEE754 представления вещественных чисел с плавающей точкой. В некоторых разделах использованы материалы свободной интернет энциклопедии Википедия.

1. Достоверность и точность математического моделирования, неэквивалентность некоторых матрично-векторных преобразований на компьютере, трудности программирования математических вычислений с вещественными числами с плавающей точкой

1.1. Источники ошибок в процессе математического моделирования

Огромное значение для вычислений, связанных с проектированием сложной и опасной техники имеют достоверность, надежность и точность компьютерных расчетов. Как показывает анализ некоторых из происшедших за последнее время аварий, одной из причин стали существенные погрешности компьютерных вычислений, результаты которых попали в конечную конструкторскую документацию. Главным недостатком вычислительных методов, используемых при проектировании и управлении и реализованных в лучших зарубежных программных системах, является возможная выдача ошибочных результатов компьютерных вычислений без предупреждения пользователей об их недостоверности. Подавляющее большинство универсальных и, особенно, специализированных программ математического моделирования технических систем и объектов разработано и разрабатывается с использованием языка Си (Си++). При этом программисты часто сами пишут программы решения систем ОДУ и СЛАУ, считая это простой задачей, а затем разрабатывают прекрасные интерфейсы и визуализацию неверных результатов решения сложных вычислительных задач.

Процесс математического моделирования технических изделий и объектов в общем случае (без использования программных комплексов математического моделирования) можно представить как последовательность четырех основных этапов:

1. Получение на основе фундаментальных физических законов исходной математической модели для объекта математического моделирования в форме систем уравнений, математических выражений, зависимостей и т.п.
2. Выбор численных методов и алгоритмов решения или выбор готового решателя (в этом случае 3-й этап не нужен) для уравнений математической модели.
3. Разработка на выбранном алгоритмическом языке и отладка расчетной программы-решателя уравнений математической модели.
4. Ввод исходных данных для разработанной или выбранной программы-решателя уравнений математической модели, получение и обработка результатов математического моделирования на компьютере.

При математическом моделировании промышленных изделий и объектов соответственно можно выделить четыре основных источника ошибок в процессе математического моделирования:

1. Погрешность исходной математической модели. Это чрезвычайно важный источник ошибок, т.к. он является неустранимым. Например, 2-ой закон Ньютона в дифференциальной форме не всегда даст корректные результаты математического моделирования, даже в отсутствии других источников ошибок математического моделирования. Однако анализ этих ошибок выходит за рамки данной книги, т.к. относится к соответствующим предметным областям математического моделирования (multi-discipline simulation).
2. Ошибки, связанные с неверным выбором численных методов и алгоритмов. Это устранимый источник ошибок, т.к. инженеру не следует самому программировать решение вычислительных задач, а следует использовать готовые программы-решатели уравнений математических моделей из математических пакетов программ или известных библиотек математических программ. Это устранимый источник ошибок математического моделирования.
3. Ошибки, связанные с ошибками программирования и ошибками вычислений из-за ограниченной разрядной сетки компьютера. Ошибки программирования являются устранимыми ошибками и их можно исправить, поэтому в данной книге анализируются только неустранимые ошибки вычислений, связанные, в частности, с ошибками округления чисел при вычислениях на компьютере и которые обязательно надо свести к минимуму или сообщать пользователю соответствующей программы о недопустимой величине этих ошибок.
4. Ошибки, связанные с вводом неверных исходных данных. Это устранимый источник ошибок математического моделирования.

Математические модели динамических процессов в реальных технических системах и объектах необходимо получать на основе фундаментальных физических законов только в форме систем ДАУ, не разрешенных относительно производных, и решать эти системы без каких-либо преобразований и без получения "правой" части для производных в явном, аналитическом виде.

При математическом моделировании динамических процессов в реальных технических системах и объектах состояние этих систем и объектов обычно рассматривают в пространстве дифференцируемых переменных или в «пространстве переменных состояния» с получением в явном аналитическом виде производных этих переменных по времени, но этого недостаточно для достоверного и точного математического моделирования реальных технических систем и объектов, так как любой дифференцируемой переменной состояния в реальном мире обычно соответствует алгебраическая переменная. Например, в электрике и электронике переменным состояния, напряжениям на емкостях и токам через индуктивности, соответствуют алгебраические переменные, токи через емкости и напряжения на индуктивно-

стях, в механике переменным состояния, скоростям тел определенной массы, соответствуют алгебраические переменные, соответствующие силы инерции, и т.д. и т.п., поэтому состояние реальных динамических систем и объектов следует рассматривать в пространстве дифференциально-алгебраических переменных.

Основным физическим свойством алгебраических переменных является возможность идеального скачка значения этих переменных в бесконечно малый отрезок времени, в то время, как для дифференцируемых переменных состояния такие скачки не возможны, поэтому общепринятое приведение систем ДАУ к системам ОДУ в нормальной форме Коши с преобразованием алгебраических переменных в дифференцируемые переменные состояния физически неверно и полученная таким образом математическая модель динамических процессов в реальных технических системах и объектах будет физически не достоверна.

Кроме того, серьезная неустранимая ошибка вычислений состоит в использовании эквивалентных (равносильных) преобразований исходных уравнений. Профессор СПбГУ Петров Ю. П. в своих книгах [1,2] показал, что общепринятые (известные со средней школы) эквивалентные преобразования систем алгебраических и дифференциальных уравнений (умножение-деление на число, не равное нулю, сокращение подобных членов уравнений, по членное аналитическое дифференцирование уравнений и т.п.), которые тотально применяются в инженерных методиках расчетов, могут существенно изменять свойства исходных систем уравнений и давать неверный результат математического моделирования реальных систем и объектов при изменении их параметров. Например, для сверхжестких и сверх колебательных классов задач математического моделирования динамических процессов в технических системах и объектах получаемое даже для номинальных (расчетных) значений параметров решение с помощью известных математических программ иногда сильно отличается от корректного, достоверного и точного решения. Сверхжесткими системами назовем системы ОДУ со степенью жесткости более 10^6, которым соответствуют СЛАУ с числом обусловленности более 10^6, а сверх колебательными системами – системы ОДУ, которые моделируют колебательные системы с добротностью более 10^6. Главным недостатком вычислительных методов, используемых для инженерного анализа динамических процессов при проектировании и управлении и реализованных в лучших зарубежных программных системах, является возможная выдача ошибочных результатов компьютерных вычислений для вышеуказанных классов задач без предупреждения пользователей об их недостоверности.

Исследования показали, что одной из причин ряда техногенных аварий и катастроф является недостоверность результатов применения некоторых широко распространенных методов компьютерных вычислений, реализованных в известных системах автоматизированного проектирования и автоматического управления. Потери для госбюджета от таких аварий и катастроф только в военной авиации России превышают 3 млрд. рублей в год [2]. Например,

авария аэробуса А-310, произошедшая в 1994 г. вблизи г. Междуреченск, произошла вследствие быстрого нарастания отклонений крена и тангажа самолета от их нормальных значений в режиме автопилота. Пока экипаж переходил на ручное управление (на месте пилота сидел его сын), отклонения возросли настолько, что ввести их в нормальные рамки уже не было возможности. Аэробус упал и разбился. Через несколько месяцев аналогичная потеря устойчивости произошла с другим аэробусом А-310 вблизи Бухареста. Также внезапно стали нарастать отклонения крена и тангажа самолета от нормальных значений. Однако на этот раз летчик сумел быстро отключить автопилот и успел в режиме ручного управления выровнять самолет. Когда после благополучной посадки стали проверять автопилот и систему управления, выявилось, что они в полном порядке и работают устойчиво. Можно сделать вывод, что система автоматического управления на аэробусе А-310 была спроектирована так, что она была способна терять устойчивость при вариациях некоторых своих параметров или комбинациях вариаций. Используемое для проектирования автопилотов программное обеспечение было основано на преобразованиях исходных систем обыкновенных дифференциальных уравнений к нормальной форме Коши и применении, казалось бы, многократно проверенных математических методов, однако достоверность преобразованных математических моделей при некоторых вариациях параметров оказалась недостаточной и даже проведение натурных испытаний оказалось недостаточным. Свойство систем управления терять устойчивость при некоторых сочетаниях параметров системы управления Петров Ю.П. назвал «скрытой параметрической неустойчивостью» систем управления.

Кроме того, многие эквивалентные преобразования, выполняемые численно, не являются эквивалентными преобразованиями из-за ограниченной разрядной сетки компьютера. Только аналитические, символьные преобразования исходных уравнений математических моделей являются строго эквивалентными.

Например, для системы линейных алгебраических уравнений эквивалентными будут только перестановки строк и столбцов матрицы коэффициентов, а матричные операции умножения и сложения уже не будут эквивалентными. Это утверждение легко обосновать примером умножения матрицы коэффициентов СЛАУ на обратную матрицу. Теоретически должна получиться единичная матрица, но на компьютере этот результат для плохо обусловленных матриц не получится, поэтому надо использовать только те методы и алгоритмы, которые не требуют численных матрично-векторных преобразований исходных уравнений. С помощью нормирования исходных данных и матрично-векторных преобразований можно всегда плохо обусловленную СЛАУ превратить в хорошо обусловленную, но полученная после таких преобразований СЛАУ из-за ограниченной разрядной сетки компьютера не будет соответствовать исходной СЛАУ, следовательно, надо разработать методы и алгоритмы достоверного и точного решения СЛАУ без каких-либо численных преобразований.

При математическом моделировании следует различать три вида погрешности математического моделирования:

1. Достоверность получаемых результатов математического моделирования, которая соответствует качественно правильным результатам математического моделирования тестовых практических и математических задач и связана с глобальной погрешностью математического моделирования.

2. Математическая точность (accuracy) результатов математического моделирования тестовых практических задач, которая связано с локальной погрешностью математического моделирования и количественно оценивает погрешность математического моделирования, обычно математическая точность указывается в процентах по отношению к известному, заведомо точному решению этих задач.

3. Компьютерная точность (precision) результатов решения тестовых математических задач, которая количественно оценивает погрешность математического моделирования и указывается в количестве верных значащих десятичных цифр в решении тестовых математических задач.

Аналитические доказательства достоверности и точности математического моделирования считаются "истиной в последней инстанции", но аналитические доказательства справедливы только на бумаге и только для не жестких и хорошо обусловленных систем ОДУ-ДАУ и СЛАУ, при реализации на компьютере ограниченная разрядная сетка сводит на нет многие доказательства. Истиной в последней инстанции следует считать только вычислительные эксперименты на компьютерах. Например, на бумаге (1/3 + 1 - 1) x 3 = 1 (последовательность арифметических операций при решении СЛАУ методом Гаусса), но после вычислений на алгоритмическом языке Си со стандартной для математических пакетов удвоенной точностью (double precision) в стандарте представления чисел IEEE-754 получим 0.99999999999999978000000000000000.

1.2. Представление чисел при компьютерных вычислениях, вещественные числа с плавающей точкой и целые числа

Множество целых чисел конечно, но мы всегда можем подобрать такое число бит, чтобы представить любое целое число, возникающее при решении конкретной задачи. Множество действительных чисел практически бесконечно, но еще оно и непрерывно, поэтому, сколько бы мы не взяли бит, мы неизбежно столкнемся с числами, которые не имеют точного представления. Вещественные числа с плавающей точкой (запятой) (floating point) — один из возможных способов представления действительных чисел, который является компромиссом между точностью и диапазоном принимаемых значений.

Сравним пример программирования задач в 3D пространстве, определённом целыми числами в 3D пространстве и определённом вещественными числами с плавающей точкой в 3D

пространстве. Для примера рассмотрим тип float языка Си, когда и целые числа, и вещественные числа с плавающей точкой хранятся в 32-х битных машинных словах. Очевидно, что существуют только 2^32 возможных комбинаций битов, следовательно, теоретически возможное число «плавающих» чисел равно числу целых. На самом деле возможное количество «плавающих» целых чисел будет немного меньше, поскольку некоторые комбинации битов являются «не числами» (“Not a Number”, NaN), но ради простоты мы этот факт будем игнорировать. Плюс к этому будем рассматривать только значения со знаком для упрощения изложения. Однако числа с плавающей точкой могут представлять значения в гораздо более широком диапазоне: от 0 до 2^128.

Принцип, по которому меньшее количество чисел может представлять большее количество значений, довольно очевиден, особенно если вы знаете способ представления вещественных чисел с плавающей точкой. Однако будет полезно ещё раз проанализировать нюансы. Между каждой степенью двойки находится одинаковое количество чисел с плавающей точкой. То есть, от 1 до 2 существует 8388608 (или 2^23) возможных плавающих чисел, и от 2 до 4 существует такое же количество чисел. Точно такое же количество чисел находится между 32768 и 65536, или между 0.03125 и 0.0625.

Можно сказать об этом и другими словами: если мы представляем линейные координаты с помощью числа с плавающей точкой, то мы имеем большее количество возможных точек между центром и точкой на расстоянии 1 миллиметр, чем возможных точек между центром и точкой на другом конце планеты. Это означает, что точность представления позиции с помощью чисел с плавающей точкой зависит от того, где вы находитесь и какие единицы используете. То есть, опять же, если число с плавающей точкой значением 1.0 представляет 1 миллиметр, то, если вы находитесь близ центра (что означает, что ваша позиция близка к 0,0,0), ваша позиция может быть представлена с точностью около 0.0000001 мм, а это очень высокая точность. Однако если вы удаляетесь от центра, то точность **начинает снижаться.** На расстоянии 1 км от центра (1000000 мм), точность падает до 0.125 мм, но это всё ещё хороший показатель. Если же вы удаляетесь на расстояние в 64 км от центра, то точность падает до 4 мм, что означает, что вы можете представлять позицию с точностью в 4 мм, то есть получить лишь четверть того разрешения, которое дают целые числа. Дальше всё становится еще хуже. Если вы путешествуете до границы пространства, которое могут представить целые числа, а это 4295 км (приблизительно дистанция от Москвы до Иркутска) или 2^32мм, то поскольку у нас есть только 23 бита мантиссы, точность падает до 2^9 или 512 мм – примерно пол метра. Итак, если использовать 32-х битные числа с плавающей точкой (тип float языка Си) для представления позиции при программировании в 3D геоинформационной системе, которая включает в себя всю континентальную часть РФ, то на одном из городов ваше

положение может быть представлено с точностью до полуметра, что уже может быть недостаточно.

Использование чисел с плавающей точкой и десятичных чисел при программировании вычислительных задач также чревато трудностями. Числа с плавающей точкой и десятичные числа ведут себя совсем не так хорошо как целые, и Вы не можете предположить, что в действительности делают вычисления с плавающей точкой, у которых "должны" быть целочисленные или точные результаты. Для вычислительных задач также не следует использовать тип float языка Си.

1.3. Форматы представления вещественных чисел с плавающей точкой

Институт инженеров по электротехнике и электронике (Institute of Electrical and Electronic Engineers, IEEE) разработал международные стандарты, которые описывают представление вещественных чисел с плавающей точкой:

- стандарт ANSI/IEEE754-1985 определяет требования к реализации двоичной плавающей арифметики;

- стандарт ANSI/IEEE754-2008 обобщает прежний стандарт ANSI/IEEE754-1985, допуская дополнительно, кроме двоичного, десятичное основание представлений мантиссы и экспоненты и произвольную длину машинного слова.

Данные стандарты, кроме форматов представления, описывают также основные арифметические действия, операции вычисления остатка от деления, квадратного корня, преобразования и т.п.

В большинстве современных платформ, таких как Intel и большинстве RISC-систем аппаратно реализована плавающая арифметика, соответствующая этим стандартам. Другие стандарты на практике не используются.

Стандарт IEEE754-1985 определяет следующие основные форматы представления вещественных чисел с плавающей точкой:

1. С одинарной точностью (соответствует типам SINGLE PRECISION (32 двоичных разряда) или REAL*4 в языке Фортран и float в языке Си);

2. С удвоенной точностью (соответствует типам DOUBLE PRECISION (64 двоичных разряда) или REAL*8 в языке Фортран и double в Си), этот тип представления принят по умолчанию во всех математических пакетах программ и стандартных математических библиотеках;

3. С расширенной точностью (условно говоря, соответствует типам REAL*10 (80 двоичных разрядов) в языке Фортран и long double в языке Си).

Формат с учетверенной точностью QUAD PRECISION (128 двоичных разряда) или REAL*16 в языке Фортран стандарт IEEE754-1985 отдельно не определяет и, поэтому для

компиляторов языка Си, использующих этот стандарт, следует создавать специальные форматы для данного представления вещественных чисел.

Формальное представление вещественных чисел с плавающей точкой в стандарте IEEE754 для любого формата точности показано на рис. 1.

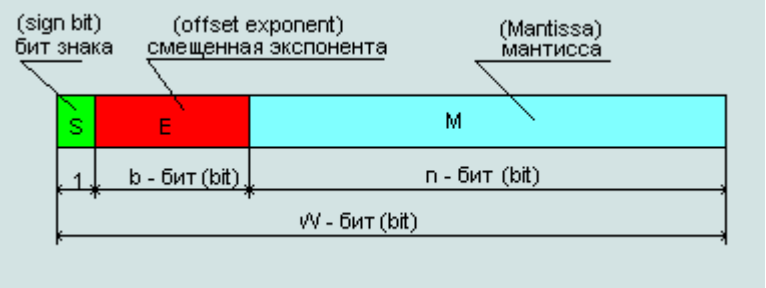


Рис. 1. Представление вещественных чисел в стандарте IEEE754

На рис.1. используются обозначения:

S - бит знака, если S=0 - положительное число; S=1 - отрицательное число;

E - смещенная экспонента двоичного числа;

M - остаток мантиссы двоичного нормализованного числа с плавающей точкой.

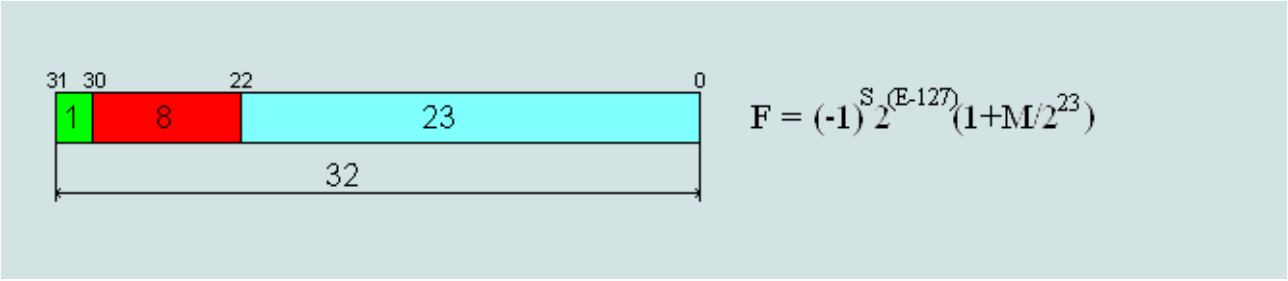
b – число бит для порядка числа;

n – число бит для мантиссы числа.

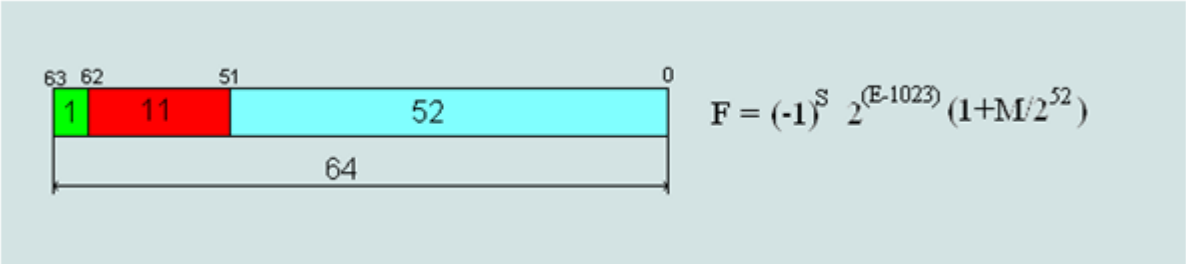
Обобщенная формула вычисления десятичных чисел с плавающей точкой из чисел, представленных в стандарте IEEE754 имеет вид:

$$F = (-1)^S \ 2^{(E-2^{(b-1)}+1)}(1+M/2^n)$$

Используя эту формулу, определяются формулы для нахождения десятичных чисел из форматов одинарной (32 бита) и удвоенной (64 бита) точности стандартов IEEE754, приведенные на рис.2 а) и 2 б) соответственно.



а)



б)

Рис. 2. Представление вещественных чисел в стандарте IEEE754 для одинарной а) и удвоенной б) точности

Отметим основные особенности представления вещественных чисел с плавающей точкой в стандартах IEEE754 при программировании математических вычислений.

В стандартах IEEE754 число «0» представляется значением с порядком, равным E=Emin+2 (для одинарной точности это -128+2=-126) и нулевой мантиссой. Введение нуля как самостоятельного числа (т.к. в нормализованном представлении нельзя представить ноль) позволило избежать многих странностей в арифметике. И хоть операции с нулем нужно обрабатывать отдельно, обычно они выполняются быстрее, чем с обычными числами. Также в стандартах IEEE754 предусмотрено представление для специальных чисел, работа с которыми вызывает исключение. К таким числам относится бесконечность (±∞) и неопред еленность (NaN). Эти числа позволяет вернуть адекватное значение при переполнении. Бесконечности представлены как числа с порядком E=Emax-1 (для одинарной точности это 128-1=127) и нулевой мантиссой. Получить бесконечность можно при переполнении и при делении ненулевого числа на ноль. Бесконечность при делении определяется, исходя из существования пределов, когда делимое и делитель стремятся к какому-то числу. Соответственно, c/0==±∞ (например, 3/0=+∞, а -3/0=-∞), так как если делимое стремится к константе, а делитель к нулю, предел равен бесконечности. При 0/0 предел не существует, поэтому результатом будет неопределенность. В стандартах IEEE754 NaN представлен как число, в котором E=Emax-1, а мантисса не нулевая. Любая операция с NaN возвращает NaN. При желании в мантиссу можно записывать информацию, которую программа сможет интерпретировать. Стандартом это не оговорено и мантисса чаще всего игнорируется. В описанном представле-

нии чисел с плавающей точкой существует два нуля, которые отличаются только знаком. Так, 3·(+0)=+0, а 3·(-0)=-0. Но при сравнении +0=-0. В стандарте знак сохранили умышленно, чтобы выражения, которые в результате переполнения или потери значимости превращаются в бесконечность или в ноль, при умножении и делении все же могли представить максимально корректный результат. Например, если бы у нуля не было знака, выражение 1/(1/x)=x не выполнялось бы верно при x=±∞, так как 1/∞ и 1/-∞ равны 0.

С ошибками из-за погрешностей округления в современной арифметике с плавающей точкой встретиться сложно, особенно если использовать удвоенную точность. Правило округления в стандартах IEEE754 говорит о том, что результат любой арифметической операции должен быть таким, как если бы он был выполнен над точными значениями и округлен до ближайшего числа, представимого в этом формате. Особенности округления в стандарте IEEE754:

1. Округление до ближайшего сделано не так как мы привыкли. Математически показано, что если 0,5 округлять до 1 (в большую сторону), то существует набор операций, при которых ошибка округления будет возрастать до бесконечности. Поэтому в IEEE754 применяется правило округления до четного. Так, 12,5 будет округлено до 12, а 13,5 – до 14.

2. Самая опасная операция с точки зрения округления в арифметике с плавающей запятой — это вычитание. При вычитании близких чисел значимые разряды могут потеряться, что может значительно увеличить относительную погрешность.

3. Для многих широко распространенных математических формул математики разработали специальную форму, которая позволяет значительно уменьшить погрешность при округлении. Например, расчет формулы «x^2-y^2» лучше вычислять, используя формулу «(x-y)(x+y)».

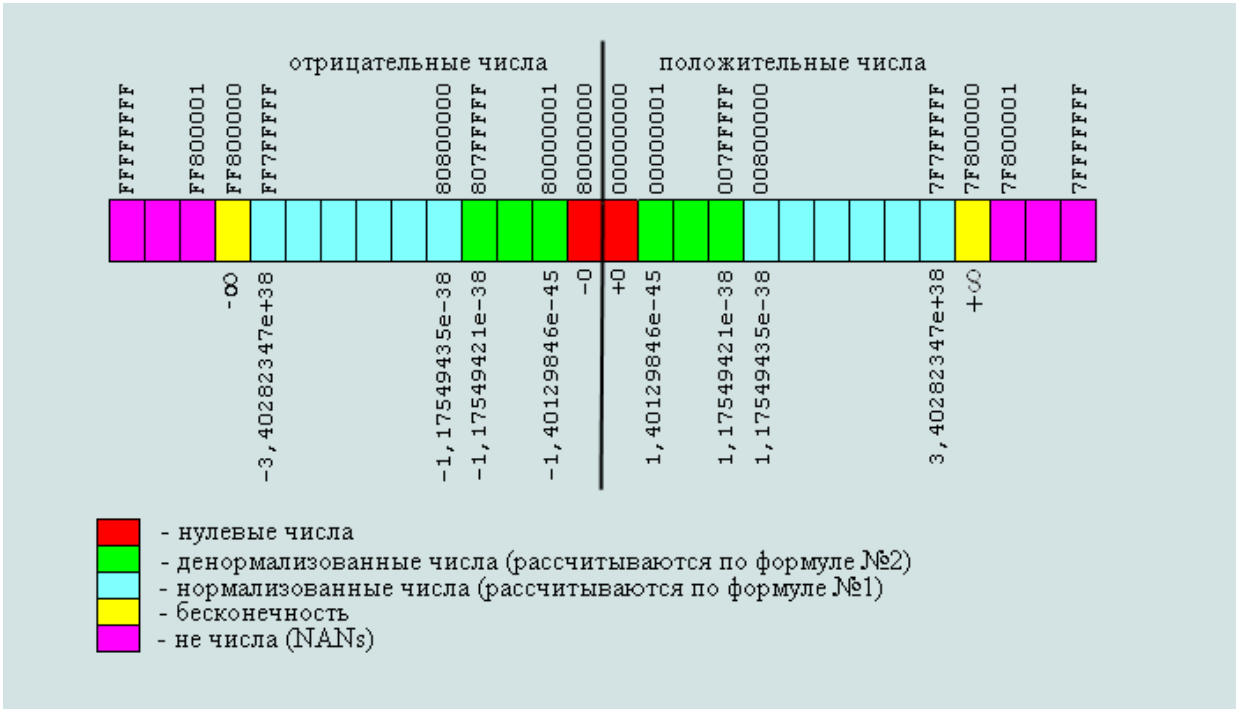
В арифметике с плавающей запятой правило (a*b)*c = a*(b*c) не выполняется для любых арифметических операций. Например, (1020+1)-1020=0 ≠ (1020-1020)+1=1.

Не все десятичные числа имеют двоичное представление с плавающей точкой. Например, число «0,2» будет представлено как «0,200000003» в одинарной точности. Соответственно, «0,2 + 0,2≈ 0,4». Абсолютная погрешность в отдельном случае может и не высока, но если использовать такую константу в цикле, можем получить накопленную погрешность.

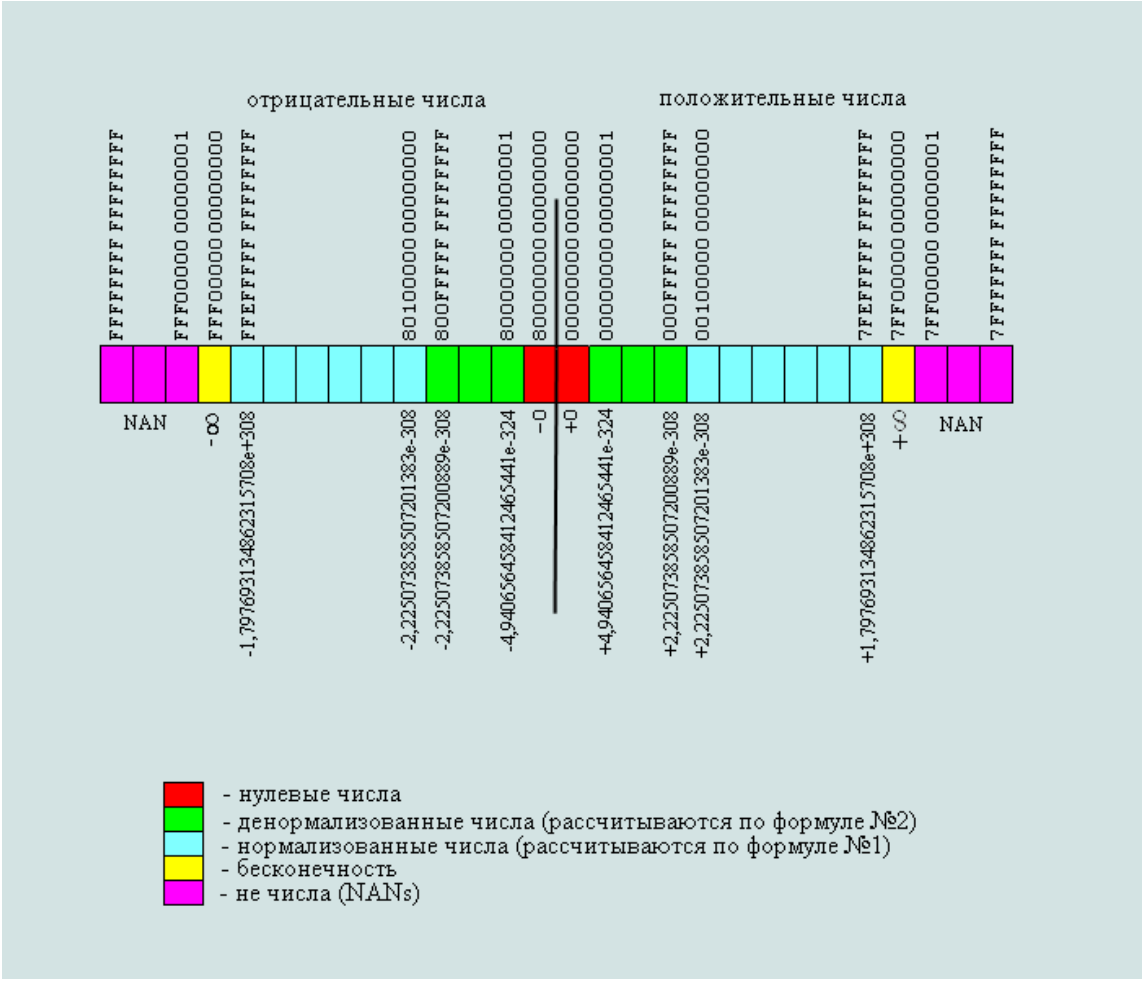
Рассмотрим представление вещественных чисел в форматах IEEE754 подробнее.

1.4. Диапазон и точность представления вещественных чисел в форматах IEEE754.

Полный диапазон чисел одинарной точности (32 бит) и удвоенной (64 бит) точности по стандарту IEEE754 показан на рис. 3.



а)



б)

Рис.3 .Диапазон чисел форматов, представленных по стандарту IEEE754 для одинарной (Single) а) и удвоенной (Double) б) точности

Числа, представленные в формате IEEE754? представляют конечное множество, на которое отображается бесконечное множество вещественных чисел. Поэтому исходное число может быть представлено в формате IEEE754 с ошибкой. Абсолютная максимальная ошибка для числа в формате IEEE754 равна в пределе половине шага чисел. Шаг чисел удваивается с увеличением экспоненты двоичного числа на единицу. То есть, чем дальше от нуля, тем шире шаг чисел в формате IEEE754 по числовой оси. Шаг чисел равен величине наименьшего разряда $\mathbf{2^{(E-22-127)}=2^{(E-149)}}$ (Single) и $\mathbf{2^{(E-51-1023)}= 2^{(E-1074)}}$ (Double). Соответственно предел максимальной абсолютной ошибки будет равен 1/2 шага числа: $\mathbf{2^{(E-150)}}$ (Single) и $\mathbf{2^{(E-1075)}}$ (Double). Относительная ошибка в % будет равна: $\mathbf{(2^{(E-150)}/F)*100\%}$(Single) и $\mathbf{(2^{(E-1075)}/F)*100\%}$ (Double).

Максимальная возможная ошибка для чисел одинарной точности приведена в таблице 1.

Таблица 1. Максимальная возможная ошибка для чисел одинарной точности

IEEE754, hex	число, dec	абсолютная ошибка, dec	относительная, %
00000001	2^{-149} ≈1,401298e-45	2^{-150}≈0,700649e-45	=50
00000002	2^{-148} ≈2,802597e-45	2^{-150}≈0,700649e-45	=25
00000032	≈7,00649e-44	2^{-150}≈0,700649e-45	=1
007FFFFF	≈1,175494e-38	2^{-150}≈0,700649e-45	≈5,96e-6
00800001	≈1,175494e-38	2^{-149} ≈1,401298e-45	≈11,9209e-6
0DA24260	≈1,0e-30	2^{-123} ≈9,4039e-38	≈9,4039e-6
1E3CE508	≈1,0e-20	2^{-90} ≈8,0779e-28	≈8,0779e-6
2EDBE6FF	≈1,0e-10	2^{-57} ≈6,9389e-18	≈6,9389e-6
3F800000	≈1,0	2^{-23} ≈1,192e-7	≈11,9209e-6
41200000	≈10,0	2^{-20} ≈9,5367e-7	≈9,5367e-6
42C80000	≈1,0e+2	2^{-17} ≈7,6294e-6	≈7,62939e-6
501502F9	≈1,0e+10	2^{10} ≈1,024e+3	≈10,24e-6
60AD78EC	≈1,0e+20	2^{43} ≈8,7961e+12	≈8,7961e-6
7149F2CA	≈1,0e+30	2^{76} ≈7,5558e+22	≈7,5558e-6
7F7FFFFF	≈+3,40282e+38	2^{104} ≈2,02824e+31	≈5,96e-6

Максимальная возможная ошибка для чисел удвоенной точности приведена в таблице 2.

Таблица 2. Максимальная возможная ошибка для чисел удвоенной точности

IEEE754, hex	число, dec	абсолютная ошибка, dec	относительная, %
00000000 00000001	2^{-1074} ≈4,940656e-324	2^{-1075}≈2,470328e-324	=50
00000000 00000002	2^{-1073} ≈9,881313e-324	2^{-1075}≈2,470328e-324	=25
00000000 00000032	≈2,470328e-322	2^{-1075}≈2,470328e-324	=1
000FFFFF FFFFFFFF	≈2,225073e-308	2^{-1075}≈2,470328e-324	≈1,110223e-14

00100000 00000001	≈2,225074e-308	2^{-1074} ≈4,940656e-324	≈2,220446e-14
2B2BFF2E E48E0530	≈1,0e-100	2^{-385} ≈1,268971e-116	≈1,268971e-14
3FF00000 00000000	=1,0	2^{-52} ≈2,220446e-16	≈2,220446e-14
54B249AD 2594C37D	≈1,0e+100	2^{280} ≈1,942669e+84	≈1,942669e-14
6974E718 D7D7625A	≈1,0e+200	2^{612} ≈1,699641e+184	≈1,699641e-14
7FEFFFFF FFFFFFFF	≈1,79769e+308	2^{971} ≈1,99584e+292	≈1,110223e-14

Из выше приведенного следует, что основная масса чисел в формате IEEE754 имеет стабильную небольшую относительную погрешность: максимально возможная относительная погрешность для числа Single составляет 2^{-23}*100% =11,920928955078125e-6 %, а максимально возможная относительная погрешность для числа Double составляет 2^{-52}*100% =2,220446049250313080847263336181 6e-14 % Точность фактических вычислений в двоичных форматах на один бит больше стандартной. Это гарантирует получение стандартной точности вычислений.

Общие сведения для чисел одинарной и двойной точности стандарта IEEE754-1985 приведены в таблице 3.

Таблица 3. Сведения о формате 32/64 бит в стандарте ANSI/IEEE754-1985

наименование формата	single-precision	double-precision
длина числа, бит	32	64
смещенная экспонента (E), бит	**8**	**11**
остаток от мантиссы (M), бит	23	52
смещение	127	1023
двоичное денормализованое число	$(-1)^S \cdot 0,M \cdot exp_2{}^{-127}$,где М-бинарное	$(-1)^S \cdot 0,M \cdot exp_2{}^{-1023}$,где М-бинарное
двоичное нормализованое число	$(-1)^S \cdot 1,M \cdot exp_2{}^{(E-127)}$,где М-бинарное	$(-1)^S \cdot 1,M \cdot exp_2{}^{(E-1023)}$,где М-бинарное
десятичное денормализованное число	$F =(-1)^S \cdot 2^{(E-126)} \cdot M/2^{23}$	$F =(-1)^S \cdot 2^{(E-1022)} \cdot M/2^{52}$
десятичное нормализованное число	$F =(-1)^S \cdot 2^{(E-127)} \cdot (1+ M/2^{23})$	$F =(-1)^S \cdot 2^{(E-1023)} \cdot (1+M/2^{52})$
Абс. макс. возм. погрешность числа	$2^{(E-150)}$	$2^{(E-1075)}$
Отн. макс. возм. погрешность денорм. числа	1/(2M)	1/(2M)
Отн. макс. возм. погрешность норм. числа	$1/(2^{23}+M)$	$1/(2^{52}+M)$
минимальное число	$\pm 2^{-149} \approx \pm 1,40129846 \cdot e^{-45}$	$\pm 2^{-1074} \approx \pm 4,94065646 \cdot e^{-324}$
максимальное число	$\pm 2^{127} \cdot (2-2^{-23}) \approx \pm 3,40282347 \cdot e^{+38}$	$\pm 2^{1023} \cdot (2-2^{-52}) \approx \pm 1,79769313 \cdot e^{+308}$

Общие сведения точности для двоичных и десятичных чисел стандарта IEEE754-2008 приведены в таблице 4.

Таблица 4. Сведения о формате 32/64 бит в стандарте ANSI/IEEE754-2008.

Имя формата	Точность	Основание	Число цифр	E min	E max	Десятичные цифры	Десятичное E max
binary16	Half precision	2	10+1	−14	+15	3.31	4.51
binary32	Single precision	2	23+1	−126	+127	7.22	38.23
binary64	Double precision	2	52+1	−1022	+1023	15.95	307.95
binary128	Quadruple precision	2	112+1	−16382	+16383	34.02	4931.77
decimal32		10	7	−95	+96	7	96
decimal64		10	16	−383	+384	16	384
decimal128		10	34	−6143	+6144	34	6144

Числа с десятичным основанием требуют использования специальных десятичных арифметико-логических процессоров.

1.5. Проблемы компьютерных вычислений, вызванные использованием стандарта IEEE754.

Стандарт IEEE754 широко применяется в технике и программировании. Большинство современных микропроцессоров изготовляются с аппаратной реализацией представления вещественных переменных в формате IEEE754. Язык программирования и программист не могут изменить эту ситуацию, иного преставления вещественного числа в микропроцессоре не существует. Когда создавали стандарт IEEE754-1985 представление вещественной переменной в виде 4 или 8 байт казалось очень большой величиной, так как объём оперативной памяти MS-DOS был равен 1 Мб. А программа в этой системе могла использовать только 0,64 Мб. Для современных ОС размер в 8 байт является ничтожным, тем не менее переменные в большинстве микропроцессоров продолжают представлять в формате IEEE754-1985. Рассмотрим ошибки компьютерных вычислений, вызванные применением чисел в формате IEEE754.

1.5.1. Ошибки связанные с точностью представления вещественных чисел в формате IEEE754.

Данная ошибка всегда присутствует в компьютерных вычислениях. Причина её возникновения описана выше. Радует только то, что величина относительной ошибки имеет размерность для одинарной точности 10^{-6} и для удвоенной точности $10^{-14.}$ Величины абсолютных ошибок могут быть значительными, максимально для одинарной точности 10^{31} и для удвоенной точности 10^{292}, что может вызывать определённые проблемы вычислений с числами, много большими 1. Это катастрофическое понижение точности вычислений в операциях, где абсолютное значение результата много меньше любого из входного значения переменных. На самом деле ошибки точности представления числа наиболее безобидные в компьютерных вычислениях, и обычно многие программисты на них не обращают никакого внимания. Тем не менее, их надо учитывать при программирования вычислительных задач.

1.5.2. Ошибки связанные с неправильным приведением типов данных.

Эти ошибки вызваны тем, что исходное число представленное в формате single и в формате double обычно не равны друг другу. Например: исходное число

123456789,123456789

Single: 4CEB79A3=+123456792,0(dec)

Double: 419D6F34547E6B75=+123456789,123456791043281555175781 25

Разница между Single и Double составит:2,876543208956718444824218 75

Поэтому переменные и промежуточные результаты компьютерных вычислений должны быть приведены к одному типу данных. Например, требование приведения данных к одному типу изложено в стандарте на язык Си ISO/IEC 9899:1999. Обратите внимание на то, что недостаточно просто привести все исходные данные к одному типу. Необходимо привести также результаты промежуточных операций к одному типу.

1.5.3. Ошибки вызванные сдвигом мантисс.

Эти ошибки связаны с потерей точности результата при неполном пересечении мантисс чисел на числовой оси. Если мантиссы чисел не пересекаются на числовой оси, то операции сложения и вычитания между этими числами невозможны. Если числа отличаются более чем в 2^{23} (для single) и 2^{52} (для double), то операции сложения и вычитания между этими числами не возможны. Максимальная относительная погрешность результата операции равна при-

мерно 5,96e-6%, что не превышает относительную погрешность представления числа. Хотя с относительной погрешностью здесь всё в порядке, здесь есть другие проблемы. Во-первых, работать с числами можно только в узком диапазоне числовой оси, где мантиссы пересекаются. Во-вторых, для каждого исходного числа существует предел выполнения цикла называемый "Циклической дырой". Если существует цикл в котором исходное число суммируется к сумме, то существует численный предел суммы для этого числа. То есть сумма, достигнув определённой величины, перестает увеличиваться от сложения её с исходным числом, такая программа будет работать в бесконечном цикле. На самом деле опытный программист никогда не будет делать циклическое вычитание(или суммирование) таким образом. Обратите внимание, что программист должен быть абсолютно готов к тому, что некоторые основные понятия математики могут не выполняться при вычислениях в формате IEEE754. Например, правила алгебраической коммутативности (a + b) + c = (a + c) + b, как было указано выше, обычно не выполняются при таких вычислениях. К сожалению, в современном фундаментальном математическом образовании этому вопросу уделяют мало внимания, оставляя решения проблем инструментария на прикладного программиста.

1.5.4. Ошибки вызванные округлением.

При компьютерных вычислениях можно выделить два вида округления:

1. Результат выполненной арифметической операции всегда округляется.

2. Вывод и ввод вещественного числа в окно Windows происходит с округлением.

В первом случае переменная округляется к ближайшему целому. При этом переменная принимает новое округлённое значение. Здесь результат операции сложения двух чисел абсолютно точен, но результат был округлен микропроцессором. Таким образом к точному результату была добавлена ошибка округления. В общем случае ошибка округления находится в пределах точности числа.

Во втором случае переменная не меняет своего значения, просто в окно Windows выводится округлённое значение вещественного числа. Получается, что исходная переменная и её отображение в окне Windows это разные числа. Это не ошибка формата IEEE754, это ошибка Windows. Для переменных типа Single в окне Windows отображается 7 значащих цифр округлённых до ближайшего целого. 3DFCD6EA =+0,123456791043281555175781 25 в окне отображается как 0,1234568 Для переменных типа Double в окне Windows отображается 15 значащих цифр округлённых до ближайшего целого. 3FBF9ADD3746F67D=+0,12345678901234609370352046653351862914860248565673828125 отображается как 0,123456789012346 Вопрос какое значение имеет переменная когда мы

вводим в окно Windows 0,123456789012346 ? Это значение будет равно такому числу: 3FBF9ADD3746F676=+0,12345678901234599655900581183232134208083152770996093 75 То есть значение переменной 3FBF9ADD3746F67D мы вообще не можем вставить напрямую в код программы. Но, мы можем схитрить и вставить в программу x=0.123456789012346 +1E-16. Полученная переменная будет равна 3FBF9ADD3746F67D. Отобразить или вести через окно ПК такое число невозможно. В результате действий Windows возникает ряд неприятных ситуаций.

1. У вас нет технической возможности отображать или вводить точные значения переменных в окнах, что само по себе очень печально.

2. Возникновение серьёзных ошибок, таких как грязный ноль. "Грязный ноль" - это когда вы или программа считаете, что переменная не равная нулю - равна нулю. В логических операциях сравнения этот не ноль может увести исполнение программы в другую ветвь алгоритма.

1.5.5. Ошибки на границе норма/денорма числа.

Эти ошибки возникают при работе с числами находящимися на границе нормализованного/денормализованного представления чисел. Они связанны с различием в представление чисел в формате IEEE754 и в различии формул перевода формата IEEE754 в вещественные числа. То есть программы должны применять различные алгоритмы в зависимости от положения вещественного числа на числовой оси формата. Кроме того, что это приводит к усложнению алгоритмов, ещё возникает неопределённость переходной зоны. Неопределённость переходной зоны заключается в том, что стандарт не определяет конкретного значения границы перехода. По сути дела граница перехода находится между двумя вещественными числами:

Последним денормализованным числом 000FFFFFFFFFFFF:

Точное десятичное значение этого числа:
+2,225073858507200889024586876085859887650423112240959465493524802562440009228235695178775888803759155264230978095043431208587738715835729182199302029437922422355981982750124204178896957131179108226104397197960400045489739193807919893608152561311337614984204327175103362739154978273159414382813627511383860409424946494228631669542910508020181592664213499660651780309507591305871984642390606863710200510872328278467884363194452 4

1586613504122347901479236958520832159762106637540161373658304419360371477835530668283453563400507407304013560296804637591858316312422452159926254649430083685186171942241764645513713542013221703137049658321015465406803539741790602258950302350193751977303094576317321085250729930508976158251915972075723245543477091246131749358028173446655273437 5e-308

и первым нормализованным числом 0010000000000000:

Точное десятичное значение этого числа: +2,225073858507201383090232717332404064219215980462331830553327416887204434813918195854283159012511020564067339731035811005152434161553460108856012385377718821130777993532002330479610147442583636071921565046942503734208375250806650616658158948720491179968591639648500635908770118304874799780887753749949451580451605050915399856582470818645113537935804992115981085766051992433352114352390148795699609591288891602992641511063466313393663477586513029371762047325631781485664350872122828637642044846811407613911477062801689853244110024161447421618567166150540154285084716752901903161322778896729707373123334086988983175067838846926092773977972858659654941091369095406136467568702398678315290680 98461721092462539672851562 5e-308

Так как граница является вещественным числом, то её точность можно задавать до бесконечности и программе может не хватить разрядности для принятия решения к какому диапазону отнести данное число.

Из сказанного выше видно, что мнение о том, что при вычислениях с плавающей точкой результат не выходит за пределы относительной погрешности представления наибольшего числа является ложным. Ошибки, перечисленные выше, складываются друг с другом. Такие ошибки могут сделать погрешность вычислений неприемлемой. Особое внимание при программировании компьютерных вычислений программист должен обратить на результаты близкие к нулю. Стандарт работает с 1985 года и полностью вошел в стандарт IEEE754-2008, который расширил точность вычислений. Но проблема достоверности компьютерных вычислений сегодня очень актуальна, и стандарт IEEE754-2008 и рекомендации ISO не решили эту проблему. Нужны инновационные идеи, но они, как правило, приходят со стороны, поэтому мнения экспертов могут быть в этом отношении ошибочны. Следует отметить новые

идеи в области представления вещественных чисел при программировании и в аппаратных средствах:

1. Аппроксиметика [5].

2. Можно представлять вещественные числа, как числа с фиксированной запятой. Для представления всего диапазона чисел Double достаточно иметь переменную состоящую из 1075 битов целой части и 1075 битов дробной части, т.е. около 270 байт на одну переменную. При этом все числа будут представлены с одинаковой абсолютной точностью. Можно работать с числами во всем диапазоне числовой оси, то есть становится возможным суммировать большие числа с малыми числами. Шаг чисел по числовой оси будет равномерен, то есть числовая ось будет линейна. Тип данных будет только один, т.е. не нужны целые, вещественные и др. типы. Здесь возникает проблема реализации регистров микропроцессоров размерностью в 270 байт, но это не проблема для современной технологии.

2. Численные методы решения систем обыкновенных дифференциальных уравнений

2.1 Разновидности обыкновенных дифференциальных уравнений

В программах решения систем ОДУ-ДАУ используются три классические постановки задач решения систем ОДУ и ДАУ:

1. Нормальная форма Коши в **пространстве переменных состояния** (явная форма):

$$dX / dt = F(X, t)$$

X - вектор координатного базиса переменных состояния размерностью m.

F - вектор-функция правых частей размерностью m.

t - – .

Заданы начальные условия $X_0 = X(0)$ и отрезок интегрирования $t=[0,TK]$, TK – заданный временной отрезок интегрирования.

2. Дифференциально-алгебраическая форма разных индексов в полном координатном базисе (полу явная форма):

$$\begin{cases} dX / dt = F(X, Y, t) \\ G(X, Y) = 0 \end{cases}$$

Y - вектор алгебраических переменных размерностью k для полного базиса.

G - вектор-функция размерностью *k.*

Заданы согласованные начальные условия: $X_0 = X(0)$, $Y_0 = Y(0)$ и отрезок интегрирования $t=[0,TK]$.

3. Дифференциально-алгебраическая форма в полном координатном базисе (неявная форма):

$$G(X, dX/dt, Y, t) = 0 \qquad (1)$$

G - вектор-функция размерностью $m + k$.

Заданы согласованные начальные условия: $X_0 = X(0)$, $Y_0 = Y(0)$ и отрезок интегрирования $t=[0,TK]$.

Задача (1) в **пространстве дифференциально-алгебраических переменных** была поставлена в полном координатном базисе Л. Петзольд в 1982 г. и ею была разработана знаменитая программа DASS (Differential Algebraic Systems Solver) на основе метода Гира с заменой производных по формуле Гира:

$$dX/dt = BDF(X, h) \ ,$$

где *BDF – Backward Differential Formula*, h – шаг интегрирования. Полученная система нелинейных алгебраических уравнений решалась методом Ньютона.

Задача решения систем ДАУ в **расширенном пространстве дифференциально-алгебраических переменных** без преобразований исходных систем ДАУ была поставлена в статье [3]:

$G(X,PX,Y,t) = 0$

$PX=dX/dt$ - вектор производных переменных состояния по времени размерностью m.

G - вектор-функция размерностью $m + k$: левая часть системы ДАУ.

Заданы согласованные начальные условия: $X_0 = X(0)$, $Y_0 = Y(0)$ и отрезок интегрирования $t=[0,TK]$.

С учетом того, что некоторые матрично-векторные преобразования на компьютере не являются эквивалентными из-за ограниченной разрядной сетки компьютеров, классические формы представления систем ОДУ следует использовать только для теоретических исследований, при условии, что все преобразования исходных дифференциальных уравнений выполняются аналитически или в символьном виде.

В теоретическом плане свойства систем ОДУ-ДАУ и методов их решения рассматриваются применительно к линейным неоднородным системам ОДУ вида:

$$dX/dt = AX + G(t), \qquad (2)$$

где A — постоянная действительная матрица размером $m \times m$, $G(t)$ известная вектор-функция времени.

Если метод интегрирования будет непригоден для решения системы (2), то он будет непригоден и для решения вышеприведенных систем. Обозначим определитель матрицы A через $\det(A)$. Спектр собственных значений — это множество всех собственных значений λ_i матрицы $A, i = 1,2,...,m$. В общем случае λ_i — комплексные величины, т. е. $\lambda_i = \mathrm{Re}(\lambda_i) + j\,\mathrm{Im}(\lambda_i)$, где $j = \sqrt{-1}$. Если собственные значения λ_i - различны, то аналитическое решение системы (3) можно представить в виде суммы отдельных фундаментальных решений однородной системы ОДУ с матрицей A и частного решения неоднородной системы (2):

$$X(t) = \sum_{i=1}^{m} c_i e^{\lambda_i t} X_i + Z(t), \qquad (3)$$

где c_i - — постоянные величины; X_i — собственный вектор матрицы A, соответствующий собственному значению λ_i; $Z(t)$ — частное решение системы (2).

Система ОДУ будет **неустойчивой**, если хотя бы для одного i имеем $\mathrm{Re}(\lambda_i) > 0$. При этом i-я компонента в (4) будет неограниченно возрастать с увеличением t. Система ОДУ будет **устойчивой**, если $\mathrm{Re}(\lambda_i) < 0$ для всех i, так как все фундаментальные решения будут иметь затухающий характер с увеличением t. Система ОДУ будет иметь **многопериодное переходное решение**, если $\mathrm{Im}(\lambda_i) >> \mathrm{Re}(\lambda_i)$ для некоторых i.

В зависимости от характера преобладающих переходных процессов для фундаментальных решений, показанных на (рис. 4. – рис. 8.), можно выделить 5 основных разновидностей фундаментальных решений, соответствующих областям комплексной плоскости λ_i, показанной на рис. 3.

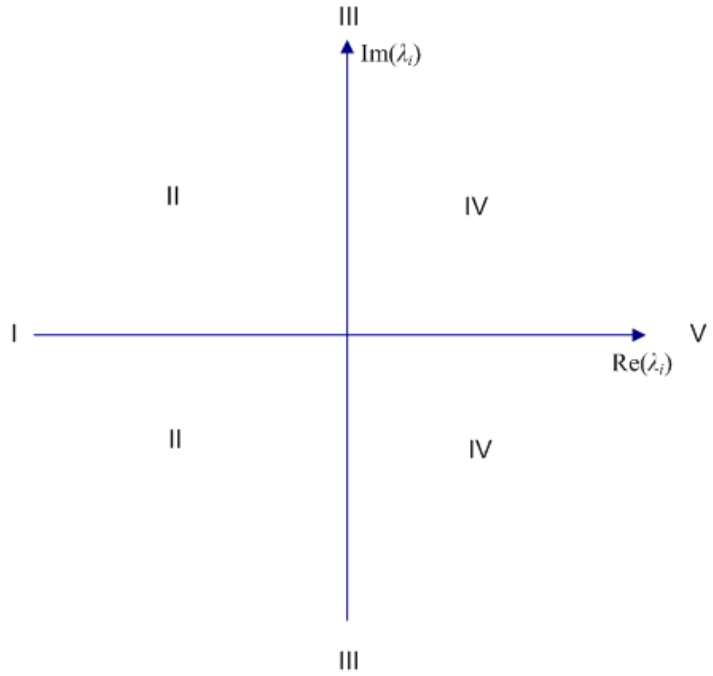


Рис. 3. Области устойчивости для 5-ти основных типов задач (I-V) при решении систем ОДУ (2)

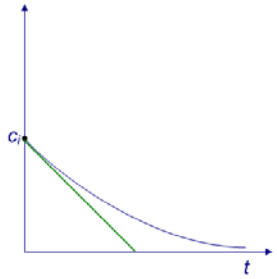


Рис. 4. Переходной процесс i-го фундаментального решения для задач типа I - фундаментальные решения лежат в основном в области I

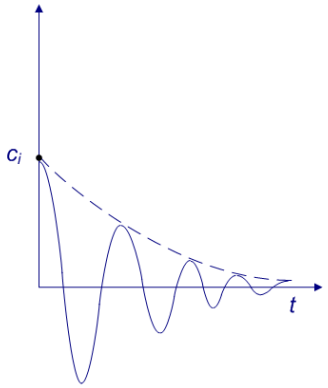


Рис. 5. Переходной процесс i-го фундаментального решения для задач типа II - фундаментальные решения лежат в основном в области II

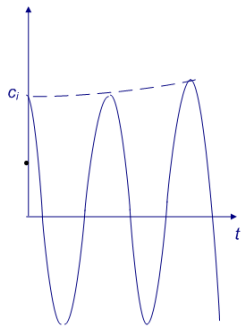


Рис. 6. Переходной процесс i-го фундаментального решения для задач типа III - фундаментальные решения лежат в основном в области III

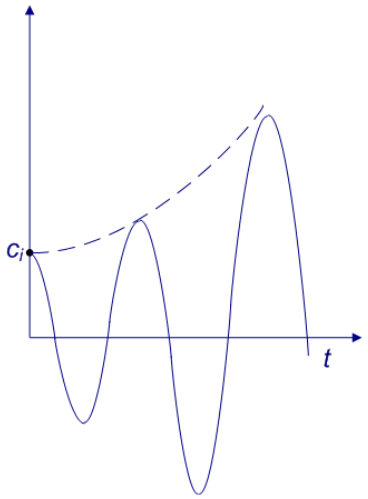


Рис. 7. Переходной процесс i-го фундаментального решения для задач типа IV - фундаментальные решения лежат в основном в области IV

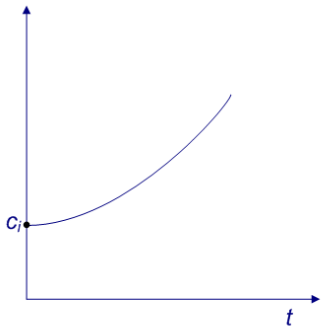


Рис. 8 Переходной процесс i-го фундаментального решения для задач типа V - фундаментальные решения лежат в основном в области V

Частное решение системы (2) определяет характер стационарного режима для устойчивых систем ОДУ при $t \to \infty$. Если $Z(t)$ — постоянный вектор при $t \to \infty$, то этот вектор определит

статическое решение системы ОДУ. Если $Z(t)$ — периодическая вектор-функция с угловой частотой $\omega = 2\pi/T$, то $Z(t)$ определит стационарное периодическое решение с периодом T.

Собственные значения λ_i характеризуют обусловленность системы ОДУ. Система ОДУ будет **плохо обусловленной** (или **жесткой**), если выполняется условие $g >> 1$, где $g = max|\lambda_i|/min|\lambda_i|$ —число обусловленности матрицы A. Обычно задачу считают жесткой, если $g > 10^6$. Для устойчивых систем ОДУ все фундаментальные решения затухают при увеличении t со скоростью, пропорциональной $\tau_i = -1/\mathrm{Re}(\lambda_i)$, называемой i-й постоянной времени системы ОДУ. Система ОДУ называется системой с большим разбросом постоянных времени, если выполняется условие $q >> 1$, где $q = max|\tau_i|/min|\tau_i|$ — разброс постоянных времени системы ОДУ. Результат решения такой системы будет содержать как быстро затухающие, так и медленно затухающие компоненты. Подобные системы возникают при математическом моделировании технических систем и объектов с физически разнородными элементами, лазерных и ядерных систем и пр. Понятия «большой разброс постоянных времени» и «плохая обусловленность» эквивалентны.

2.2. Общий алгоритм численного решения систем ОДУ-ДАУ

Рассмотрим задачу решения систем ДАУ общего вида, не разрешенных относительно производных, в следующей постановке:

$$F(X, PX, Y, t) = 0, \qquad (2)$$

где X - вектор дифференцируемых переменных размерностью m; PX - вектор производных этих переменных по переменной t размерностью m, т.е. $PX = dX/dt$; Y - вектор алгебраических переменных размерностью k; t - независимая переменная (обычно – время); F - вектор-функция размерностью l, где $l = m + k$. Заданы отрезок интегрирования и начальные условия для вектора дифференцируемых переменных: $X0=X(t0)$, где $t0$ - начальный момент времени интегрирования. Начальные значения остальных переменных, т.е. PX_0 и Y_0 рассчитываются перед началом интегрирования автоматически. Необходимо вычислить вектора $X(t)$, $PX(t)$, $Y(t)$ как функции времени на заданном отрезке интегрирования с гарантированной относительной точностью не менее 0.001.

3. Численные методы решения систем нелинейных алгебраических уравнений

4. Численные методы решения систем линейных алгебраических уравнений

5. Цикл лабораторных работ

ЛИТЕРАТУРА.

1. Ю.П. Петров Обеспечение достоверности и надежности компьютерных расчетов. – СПб.: БХВ-Петербург, 2008. - 160 с.

2. Ю. П. Петров Записки профессора. – СПб.: БХВ-Петербург,2011. – 176 с.

3. Норенков И.П., Трудоношин В.А., Федорук В.Г. Метод формирования математических моделей для адаптируемых программных комплексов анализа радиоэлектронных схем / Радиотехника, 1986, № 9. С.67-72.

4. Андронов А.В., Жук Д.М., Кожевников Д.Ю., Маничев В.Б. Библиотека математических программ-решателей на языке Си: SADEL. // http://pa10.ru, 2010.

5. Юровицкий В.М. АППРОКСИМЕТИКА: математическая теория и компьютеринг приближенных чисел. // http://www.yur.ru/science/computer/appro/monografia.htm , 1998.