БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Факультет информационных технологий и робототехники
Кафедра «Системы автоматизированного проектирования»

СОГЛАСОВАНО
Заведующий кафедрой
_____ А.В.Бородуля
27.09. 2018 г.

СОГЛАСОВАНО
Декан факультета
_____ Е.Е.Трофимченко
27.09. 2018 г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

«Алгоритмическая реализация численных методов»

для специальности 1- 31 81 12 «Прикладной компьютерный анализ данных»

Составители: Волков Василий Михайлович, Ковалева Ирина Львовна,
Напрасников Владимир Владимирович

Рассмотрено и утверждено
на заседании совета факультета информационных технологий и
робототехники 27.09. 2018 г., протокол N 1

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Цели ЭУМК: ЭУМК предназначен для изучения дисциплины «Алгоритмическая реализация численных методов» на второй ступени высшего образования по специальности 1-31 81 12 «Прикладной компьютерный анализ данных». ЭУМК содержит набор методических материалов по этой дисциплине.

Особенности структурирования и подачи учебного материала: ЭУМК состоит из четырех частей.

Теоретический раздел содержит набор методических материалов по предмету: кратких лекционных материалов, посвященных изложению в наглядном виде основных понятий и определений, презентаций и ссылок на электронные образовательные ресурсы.

Практический раздел содержит задания для проведения лабораторных занятий, а также программные коды примеров (скрипты Matlab), рассмотренных в теоретическом разделе. Данные программные коды могут быть использованы для выполнения индивидуальных заданий.

Раздел контроля знаний содержит вопросы для организации текущего контроля знаний.

Вспомогательный раздел содержит программу дисциплины, список рекомендуемой литературы.

Рекомендации по работе с ЭУМК:

- ЭУМК представлен .pdf-файлом;
- требования к системе: IBM PC-совместимый ПК стандартной конфигурации, Adobe Reader. Программа работаем в среде Windows XX;
- открытие ЭУМК производится посредством запуска файлов с расширением .pdf – в Adobe Reader

Оглавление

1. ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

1.1. Материалы к лекциям

Раздел 1. Численные методы для обыкновенных дифференциальных уравнений

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*
2. *Численные методы для обыкновенных дифференциальных уравнений [Электронный ресурс]: учебно-методическое пособие для магистрантов специальности 1-31 81 12 «Прикладной компьютерный анализ данных»/ В.М.Волков, И.Л.Ковалева – БНТУ, 2016.*

Тема 1.1. Задача Коши

Постановка задачи.

Задачи с начальными условиями (задачи Коши) широко используются для моделирования динамических систем. Существование и единственность решения задачи Коши ассоциируется с принципом детерминизма. Как правило, если уравнения и начальные условия сформулированы корректно на физическом уровне строгости, то и с математической точки зрения задача также является корректной в смысле существования и единственности решения. Число дополнительных начальных условий для однозначного определения обычно совпадает с количеством уравнений системы.

Постановка задачи Коши для системы обыкновенных дифференциальных уравнений (ОДУ) первого порядка может быть выполнена следующим образом:

$$\frac{d\mathbf{u}}{dt} = \mathbf{f}(t, \mathbf{u}), \quad t \in [0, T], \tag{1.1}$$

$$\mathbf{u}(0) = \mathbf{u_0}, \tag{1.2}$$

где

$$\mathbf{u} = \mathbf{u}(t) = [u_1(t), u_2(t), \dots u_N(t)]^T,$$

$$\mathbf{f}(t, \mathbf{u}) = [f_1, f_2, \dots, f_N]^T,$$

$$f_k = f_k(t, u_1(t), u_2(t), \dots u_N(t)), \quad k = 1, \dots, N,$$

$$\mathbf{u_0} = [u_1^0, u_2^0, \dots, u_N^0]^T.$$

где $u_k = u_k(t)$ – это искомые функции одной переменной, а f_k - заданные функции $N + 1$ переменных.

Метод Эйлера. Явные и неявные схемы. Понятие устойчивости.

Метод Эйлера - самый элементарный метод численного интегрирования задачи Коши для обыкновенных дифференциальных уравнений. Формула (1.3) известна как метод Эйлера

$$U(t_{k+1}) = U(t_k) + f(t_k, U(t_k)); \;\; t_k = k\tau \,; \; k = 0, 1, 2 \,, \;\; \dots \qquad (1.3)$$

Решение в методе Эйлера вычисляется непосредственно по явной формуле. Данный класс алгоритмов принято называть явными. Будем также называть метод Эйлера одношаговым, подчеркивая тем самым то, что для вычисления нового значения U_{k+1} при $t = t_{k+1}$ используется значение решения только в одной предыдущей точке при $t = t_k$ (на дистанции одного шага от искомого решения в новой точке).

Устойчивость численных методов для решения задачи Коши (1.1) и (1.2) можно определить требованием выполнения оценки

$$|U_k| \leqslant C_1 |U_0\| \; + C_2 \max_{0<m<k-1} |f(t_m, U_m)|, \qquad (1.4)$$

где C_1 и C_2 - положительные постоянные, не зависящие от τ , $U_k = U(t_k)$.

Данное неравенство означает, что приближенное решение непрерывно зависит от входных данных. Численный метод, который устойчив при выполнении некоторых ограничений на шаги дискретизации, называется условно устойчивым. Если для устойчивости метода никаких ограничений на шаги сетки не требуется, то такой метод называют безусловно устойчивым.

Пример 1.1. Рассмотрим пример, демонстрирующий типичные проявления эффекта потери устойчивости, когда условия устойчивости оказываются нарушенными. Для этого рассмотрим явный метод Эйлера применительно к модельному уравнению вида

$$\frac{du}{dt} = \lambda u, \quad \lambda < 0. \tag{1.5}$$

$\lambda = 1{:}5$ с начальными условиями $u(0) = 2$. Программная реализация приведена в программном блоке, результаты численных экспериментов представлены на рис.1.1

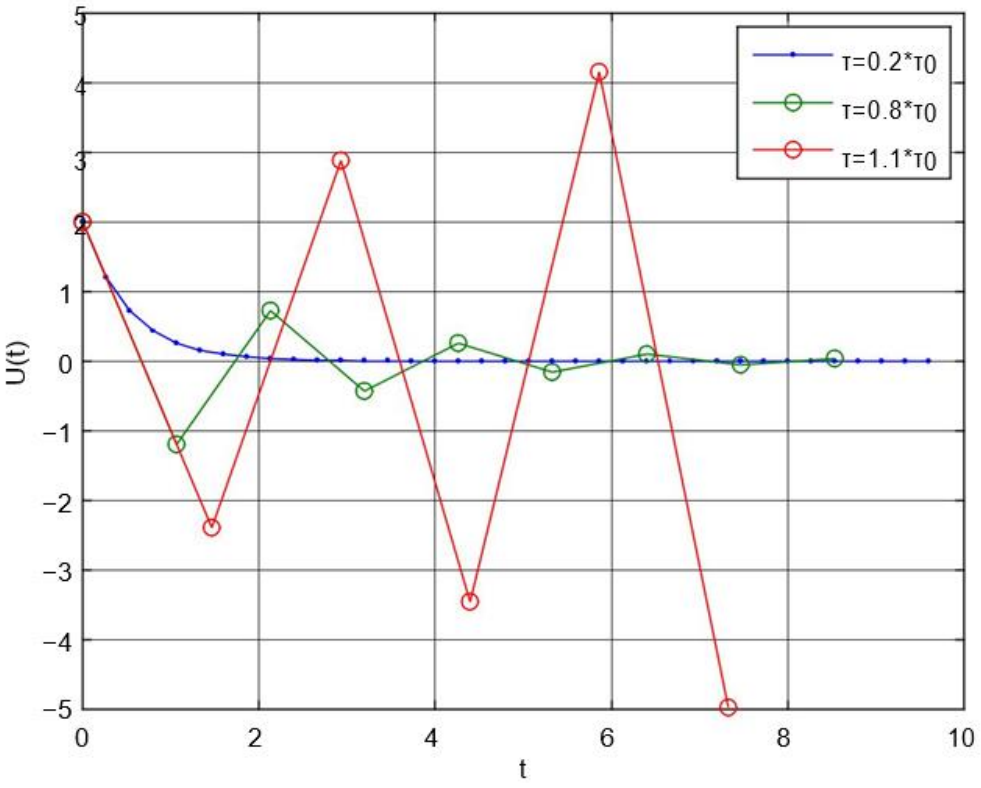


Рис.1.1 Потеря устойчивости явного метода Эйлера

Методы Рунге-Кутта. Адаптация шага сетки.

Методы Рунге – Кутты представляют собой семейство методов численного анализа задач Коши для систем обыкновенных дифференциальных уравнений вида (1.1)–(1.2). Наибольшей распространение в практике вычислений получили явные одношаговые методы Рунге – Кутты следующего вида:

$$U_{k+1} = U_k + \sum_{m=1}^{s} b_m K_m, \tag{1.6}$$

где *s*- целая постоянная, определяющая количество стадий вычислений, связанных пересчетом функции правой части, в пределах одного шага численного интегрирования,

$$\begin{aligned} K_1 &= \tau f(t_k, U_k), \\ K_2 &= \tau f(t_k + c_2\tau, U_k + a_{21}K_1), \\ &\cdots\cdots\cdots\cdots \\ K_s &= \tau f(t_k + c_s\tau, U_k + \sum_{i=1}^{s-1} a_{si}K_i). \end{aligned} \tag{1.7}$$

Коэффициенты b_m, c_m *и* a_{km} определяются из условия максимального порядка малости локальной погрешности для заданного числа стадий. Вычислительная сложность метода Рунге – Кутты зависит от количества вычислений функции *f(t;u)* и растет пропорционально числу стадий *s*, поскольку на каждой стадии значение функции вычисляется один раз.

Для *s* = 1 одностадийный метод Рунге – Кутты полностью совпадает с методом Эйлера.

Семейство методов Рунге – Кутты, относящихся к классу одношаговых явных методов обладает гибкостью в использовании и достаточно высокой эффективностью применительно к широкому кругу дифференциальных задач. Тем не менее, увеличение скорости сходимости выше четвертого порядка в методах Рунге – Кутты сопряжено с некоторым дополнительным ростом вычислительных затрат, что существенно снижает эффективность

одношаговых методов Рунге – Кутты. Такого рода недостатки можно преодолеть, используя многошаговый подход к построению дискретной модели дифференциальной задачи.

Многошаговые методы. Жесткие системы. Метод Гира.

Линейные многошаговые методы для решения задачи Коши (1.1)–(1.2) имеют следующий общий вид:

$$U_k + a_1 U_{k-1} + \ldots + a_m U_{k-m} = \tau \left[b_0 f_k + b_1 f_{k-1} + \ldots + b_m f_{k-m} \right], \quad (1.8)$$

где $f_k = f(t_k, U_k)$, $U_k = U(t_k)$, а значения коэффициентов a_n *и* b_n определяются из условий минимума локальной погрешности метода.

В качестве многошаговых методов, способных обеспечивать устойчивость, можно отметить семейство методов Адамса. Основное преимущество методов Адамса состоит в том, что, в отличие от методов Рунге – Кутты, функция правой части задачи на каждом шаге вычисляется всего один раз, независимо от порядка точности метода. Как следствие, вычислительная сложность многошаговых методов практически не зависит от порядка точности, что выгодно отличает их от одношаговых аналогов. Так, например, в методе Рунге – Кутты четвертого порядка точности функция правой части вычисляется четыре раза на каждом шаге, в то время как в явном методе Адамса можно ограничиться однократным вычислением практически для произвольного порядка точности. Таким образом, преимущество многошаговых методов становится более существенным при использовании формул более высокого порядка точности и особенно в случаях, когда вычисление функции правой части задачи (1.1)–(1.2) сопряжено со значительными вычислительными затратами.

Пример 1.2. В примере, рассмотренном ниже, выполнялось сравнение эффективности методов Рунге – Кутты и многошаговых методов Адамса, оценивая вычислительные затраты для получения заданной точности. В качестве тестовой задачи было рассмотрено уравнение

$$\frac{dz}{dt} = v_z, \quad \frac{dv_z}{dt} = -\omega^2 z,$$

для которого известно точное решение. Сравнивалась эффективность методов четвертого порядка точности. Программная реализация алгоритмов приведена в программном блоке, а результаты численных экспериментов представлены на рис.1.2.

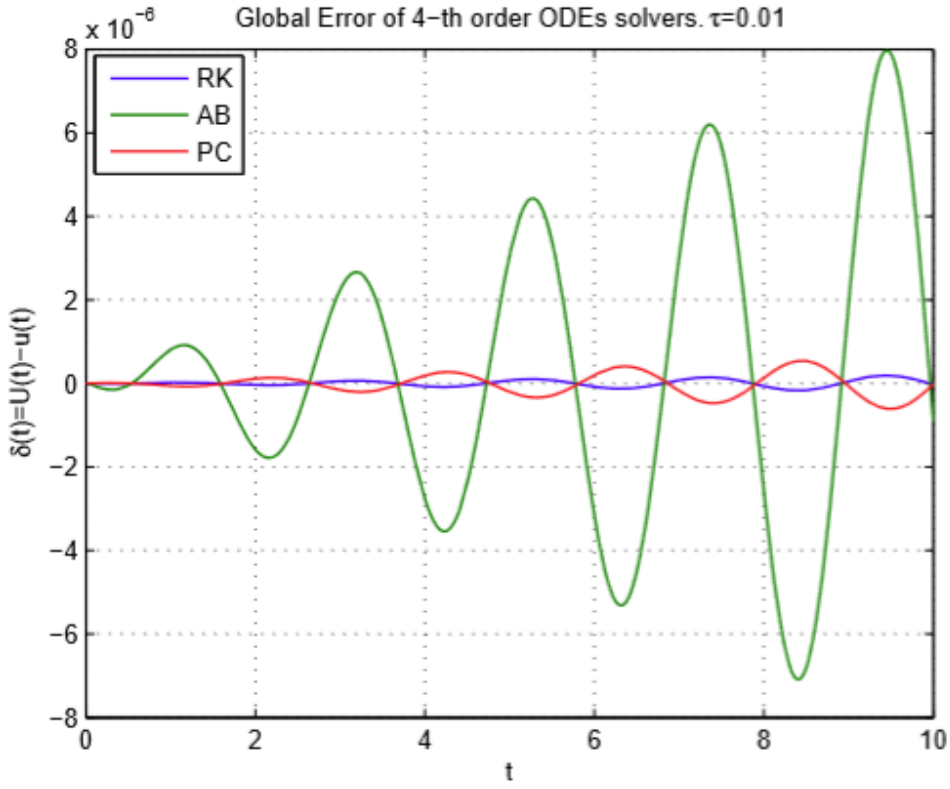


Рис.1.2. Динамика изменения погрешности методов четвертого порядка точности Рунге-Кутты (RK), Адамса – Башфорта (AB), предиктор – корректор (PC)

Большинство методов, используемых для решения задачи Коши являются условно устойчивыми. Условия устойчивости налагают определенные ограничения на размер шага численного интегрирования. Как правило, при решении задачи Коши для одного уравнения ограничения на размер шага, вытекающие из условий устойчивости, не являются более жесткими, нежели ограничения, продиктованные требованиями точности. Как следствие, при численном интегрировании одного уравнения явные методы представляются

более предпочтительными. Однако в случае систем ОДУ может возникать иная ситуация.

Для того, чтобы избежать дополнительных ограничений на размер шага, не связанных с требованиями точности, следует использовать безусловно устойчивые (А-устойчивые) неявные методы. Каждый шаг неявных методов требует больших вычислительных затрат, однако, благодаря возможности использовать более крупный размер шага без потери устойчивости, такие методы в ряде случаев оказываются более предпочтительными по сравнению с явными. В рамках неявной многошаговой схемы Адамса – Моултона можно получить абсолютно устойчивый метод, не превосходящий второго порядка точности.

Для улучшения устойчивости многошагового метода более перспективным представляется использование чисто неявной схемы, основанной на формулах дифференцирования назад (backward differentiation formulae), (backward differentiation formulae (BDF)). Данный класс многошаговых методов известен также как методы Гира.

Другой подход к решению жестких систем основан на использовании неявных методов Рунге – Кутты. Среди достоинств неявных методов Рунге – Кутты, прежде всего следует отметить их преимущества в устойчивости, что делает весьма привлекательным использование этого класса методов для численного анализа жестких систем. Следует также отметить, что порядок точности неявного метода Рунге – Кутты при одинаковом числе стадий превосходит точность явной схемы.

Пример 1.3. Сравним область устойчивости явного и неявного метода Рунге – Кутты четвертого порядка точности. Результаты численных расчетов области устойчивости методов Рунге – Кутты представлены на рис.1.3. Код программы приведен в программном блоке.

Как видно из примера, неявный метод Рунге – Кутты четвертого порядка имеет неограниченную область устойчивости, покрывающую всю левую

полуплоскость комплексной плоскости, т.е. данный метод является A-устойчивым.

Несмотря на превосходные свойства устойчивости и высокую точность неявные методы Рунге – Кутты имеют серьезный недостаток, связанный с высокой вычислительной сложностью реализации. По этой причине практический интерес привлекают преимущественно варианты неявной схемы невысокого порядка точности для решения жестких систем при умеренных требованиях к точности результатов. В частности, некоторые варианты неявных схем реализованы в функциях MATLAB1 ode23tb, ode23t и ode23s. Многошаговые методы переменного порядка, включая чисто неявные формулы дифференцирования назад, реализованы в функции ode15s.

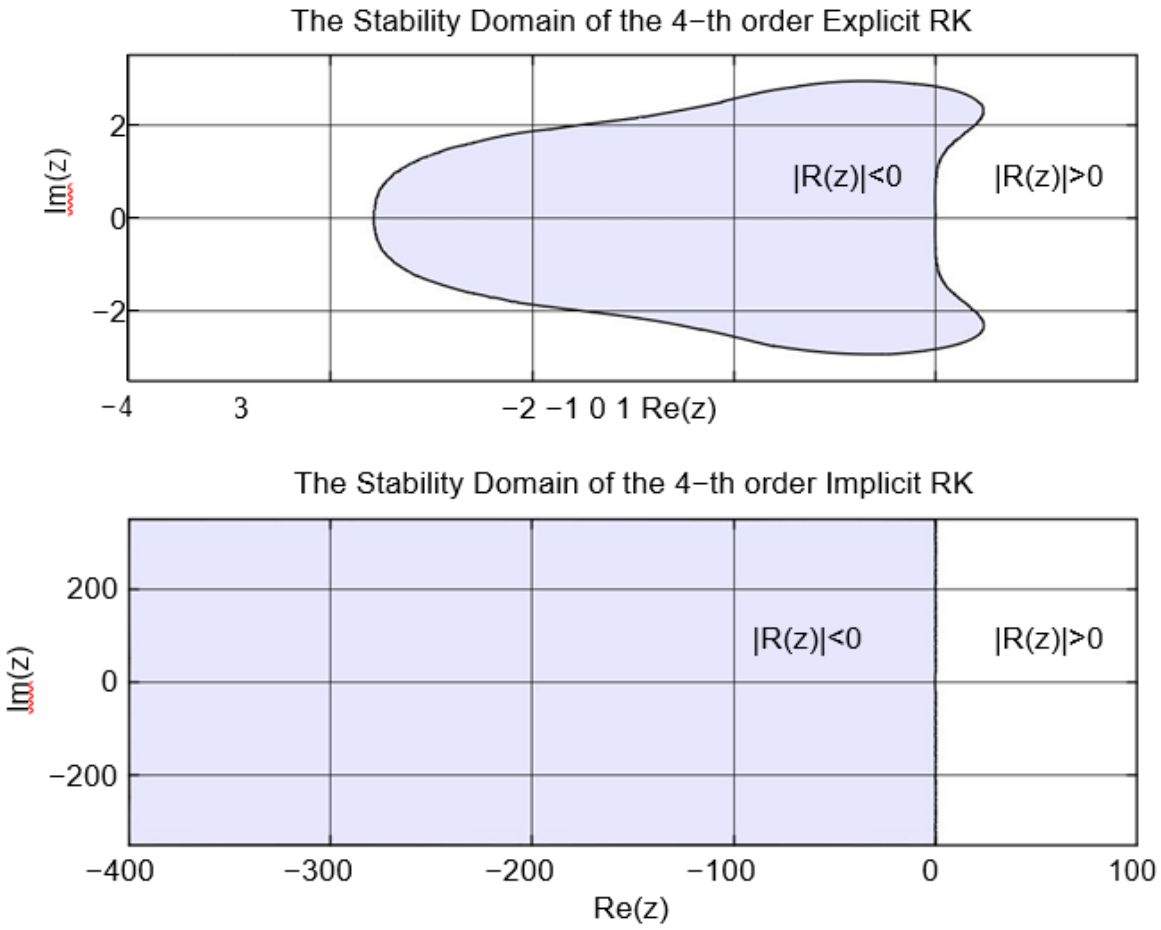


Рис.1.3. Области устойчивости явного и неявного методов Рунге – Кутты четвертого порядка точности.

Тема 1.2. Краевые задачи

Постановка задачи.

При математическом моделировании инженерно-физических задач возникает необходимость определить дополнительные условия для решения систем дифференциальных уравнений при различных значениях независимой переменной. Для простоты рассмотрим пример дифференциального уравнения второго порядка

$$\frac{d^2u}{dx^2} + a(x)u = f(x), \quad x \in (0,1), \tag{1.9}$$

с нулевыми краевыми условиями на концах отрезка:

$$u(0) = 0, \quad u(1) = 0. \tag{1.10}$$

Задачи такого типа, когда дополнительные условия задаются в разных точках, называются краевыми задачами.

Задача (1.9)–(1.10) допускает эквивалентное представление в виде системы двух дифференциальных уравнений первого порядка:

$$\begin{cases} \dfrac{dv}{dx} + a(x)u = f(x), \\ \dfrac{du}{dx} = v, \end{cases} \tag{1.11}$$

Формулировка краевой задачи в виде системы дифференциальных уравнений первого порядка в некоторых случаях представляется предпочтительной, особенно если искомые функции такой системы наделены конкретным смыслом.

Основная идея большинства методов численного анализа дифференциальных уравнений состоит в переходе от рассмотрения непрерывной задачи в бесконечно-мерном функциональном пространстве V к дискретной задаче, формулируемой в конечномерном векторном пространстве

V_N . Среди наиболее универсальных подходов к решению краевых задач для дифференциальных уравнений следует в первую очередь отметить методы конечных разностей и конечных элементов.

Метод конечных разностей и конечных элементов для решения краевых задач.

В методе конечных разностей, в отличие от исходной дифференциальной задачи, искомое решение является не функцией, заданной в некоторой ограниченной области, а множеством значений этой функции, определенных в конечном числе точек, принадлежащих области определения решения. Конечно–разностный метод основан на замене производных дифференциальной задачи на соответствующие разностные приближения.

Теоретическую основу метода конечных элементов составляют слабая формулировка задачи, метод Галеркина и представление решения в виде линейной комбинаций финитных базисных функций.

Основное отличие метода конечных элементов от разностных методов состоит в так называемой слабой формулировке дифференциальной задачи. Слабая формулировка задачи получается путем умножения дифференциального уравнения (1.12) на некоторую дифференцируемую тестовую функцию $\Psi(x)$ с последующим интегрированием полученного равенства, используя формулы интегрирования по частям:

$$\int_0^1 \lambda(x)\frac{du}{dx}\frac{d\Psi(x)}{dx}\,dx + \int_0^1 q(x)u(x)\Psi(x)\,dx + \int_0^1 f(x)\Psi(x)\,dx = 0. \tag{1.12}$$

Несмотря на принципиальное отличие методов конечных разностей и конечных элементов, построенные в рамках данных подходов дискретные модели могут при определенных условиях совпадать.

Применение разностных методов и метода конечных элементов при решении линейной краевой задачи для уравнения второго порядка приводит к дискретным моделям в виде системы линейных алгебраических уравнений с

разреженной ленточной матрицей трехдиагонального вида. Такого рода системы могут быть эффективно реализованы с помощью метода Гаусса или его модификаций метода прогонки. Вычислительная сложность реализации такой системы имеет порядок $O(N)$ вместо $O(N^3)$, имеющего место в случае численного анализа линейных систем с полной матрицей.

При решении нелинейных дифференциальных задач соответствующая дискретная модель также будет нелинейной. В этом случае для анализа алгебраической задачи необходимо использовать подходящий итерационный метод, выбор которого чаще всего зависит от постановки исходной задачи.

Эффективность методов конечных разностей и конечных элементов во многом связана со структурой получаемых при этом дискретных моделей в виде систем с разреженной матрицей ленточного типа, допускающей эффективное обращение. Как следствие, методы конечных разностей и конечных элементов во многом отвечают компромиссу между достаточно высокой точностью и умеренной вычислительной сложностью.

Эти методы легко обобщаются на случай неоднородных и адаптивных сеток, что важно при моделировании задач с неоднородными решениями, включая сингулярные режимы и пространственно-локализованные особенности поведения. За последние десятилетия разработано большое число модификаций данных методов, которые позволяют улучшить их вычислительные характеристики как в общем случае, так и в случае решения важных частных задач. Достоинства и недостатки этих двух классов методов могут быть кратко резюмированы следующим образом: если вам необходимо решить дифференциальную краевую задачу как можно быстрее, а вы не имеете опыта в области численного анализа, тогда разностный метод, пожалуй, может быть лучшим выбором. Однако, если вы ищите мощный вычислительный аппарат, который позволит решить не только текущие проблемы, но способен обеспечить надежную поддержку при развитии и совершенствовании дифференциальной модели, тогда стоит обратить внимание на метод конечных элементов. Преимущества метода конечных элементов проявляется в еще

большей степени при решении дифференциальных задач с частными производными.

Спектральные методы.

Идея спектральных методов состоит в том, что искомое приближенное решение представляется в виде линейной комбинации некоторого множества базисных бесконечно дифференцируемых функций $\phi_n(x)$, $n = 0,1,2, \ldots, N\text{-}1$.

$$u(x) \approx \tilde{u}(x) = \sum_{n=0}^{N-1} a(n)\psi_n(x), \quad x \in [a,b], \tag{1.13}$$

где коэффициенты $a(n)$ определяются таким образом, что разность между искомой функция $u(x)$ и ее приближенным представлением $\tilde{u}(x)$ была минимальной в определенном смысле. В отличие от метода конечных элементов, для спектральных методов не требуется ограниченности носителя базисных функций.

Типичным примером спектральных методов является метод Фурье, основанный на использовании тригонометрических базисных функций, периодических на некотором интервале $x \in [0,L]$.

$$\psi_n(x_k) = \exp\left(\frac{i2\pi n}{L}x_k\right), \quad x_k = kh, \quad k = 0,1,2,\ldots N-1, \quad h = L/N. \tag{1.14}$$

Вычислительная сложность спектральных методов существенно выше по сравнению с разностными и конечно элементными. Тем не менее, высокая вычислительная сложность спектральных моделей в случае достаточно гладких решений компенсируется высокой точностью, что обеспечивает в итоге преимущества данного класса методов.

Пример 1.4. Рассмотрим пример спектрального дифференцирования с использованием метода Фурье, применительно к функциям, имеющим различную степень гладкости:

$$u_1(x) = \exp(-16x^4), \quad u_2(x) = \begin{cases} \exp(-16x^2), & x \in [-\pi, 0], \\ \exp(-16x^4), & x \in [0, \pi]. \end{cases}$$

Функция $u_1(x)$ является бесконечно дифференцируемой, в то время как функция $u_2(x)$ имеет лишь одну непрерывную производную. Результаты численных экспериментов представлены на рис.1.4. Программная реализация спектрального дифференцирования Фурье приведена в программном блоке.

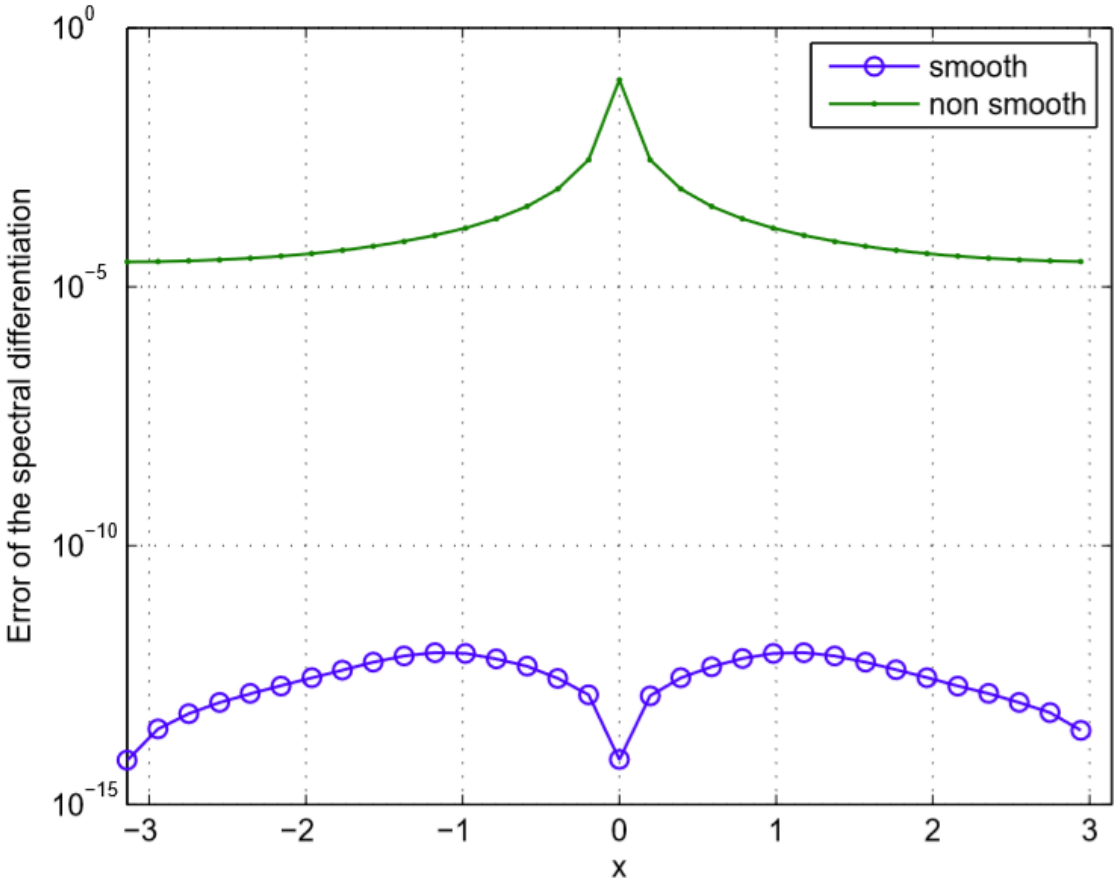


Рис.1.4. Погрешности спектрального дифференцирования Фурье в случае гладкой и не гладкой функции при N = 128

Представленные на рис. 1.4 результаты показывают существенную зависимость погрешности спектрального дифференцирования Фурье от гладкости дифференцируемой функции. Для бесконечно дифференцируемой функции $u_1(x)$ погрешность спектрального дифференцирования достигает

порога вычислительной погрешности на сравнительно грубой сетке $N >= 128$. В случае функции $u_2(x)$ производные высшего порядка разрывные при $x = 0$. Как следствие, ошибка спектрального дифференцирования в последнем случае существенно выше и медленно убывает с уменьшением шага сетки.

Наряду с методом Фурье, одним из наиболее эффективных методов решения дифференциальных задач является спектральный метод Чебышева, который может использоваться для произвольных краевых условий.

Раздел 2. Численные методы для дифференциальных уравнений с частными производными

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*

Тема 2.1. Метод конечных разностей

Нестационарное одномерное уравнение теплопроводности.

В отличие от обыкновенных дифференциальных уравнений, дифференциальные уравнения с частными производными обеспечивают дополнительные возможности моделирования реальных прикладных задач, решения которых описываются функциями многих переменных. В большинстве случаев задачи для уравнений с частными производными могут рассматриваться как естественное обобщение соответствующих краевых задач и задач Коши для обыкновенных дифференциальных уравнений. Одним из типичных примеров такого типа обобщений может служить зависящее от времени уравнение теплопроводности, которое является нестационарным аналогом обыкновенного дифференциального уравнения, рассмотренного выше:

$$\frac{\partial u}{\partial t} - \frac{\partial u}{\partial x}\lambda(x)\frac{\partial u}{\partial x} - q(x,t)u = f(x,t), \quad (x,t) \in \Omega = [0,L] \times [0,T]. \tag{2.1}$$

Задачи для уравнения (2.1) подразумевают формулировку соответствующих начальных и граничных условий:

$$u(x, t = 0) = u_0(x), \tag{2.2}$$

$$\left[\alpha_0(x,t)\frac{\partial u}{\partial x}+\beta_0(x,t)u+\gamma_0(x,t)\right]_{x=0}=0,$$

$$\left[\alpha_L(x,t)\frac{\partial u}{\partial x}+\beta_L(x,t)u+\gamma_L(x,t)\right]_{x=L}=0. \tag{2.3}$$

Согласованность, устойчивость и сходимость.

Ключевой вопрос относительно любого численного метода касается прежде всего его точности. Погрешность разностного метода оценивается на основе разности между точным решением дифференциальной задачи, которое в общем случае неизвестно, и приближенным решением дискретной задачи в узлах сетки. Оценка погрешности может быть получена на основе численных экспериментов или теоретически. Теоретические оценки погрешности разностных схем для линейных дифференциальных задач возможны на основе исследования согласованности (аппроксимации) и устойчивости разностных схем.

Считается, что разностная схема сходится если норма погрешности, которая определяется разностью между точным и приближенным решениями в узлах сетки, стремится к нулю при $\tau \to 0$, $h \to 0$:

$$\lim_{\tau,h\to 0}\|U-u\|=0. \tag{2.3}$$

Как правило, норма погрешности связана с шагами дискретизации τ и h оценками вида

$$\|U-u\|=O(h^s+\tau^p), \quad \text{или} \quad \|U-u\|<C_1h^s+C_2\tau^p, \tag{2.4}$$

где числа s и p характеризуют скорость сходимости или порядок точности, C_1 и C_2 — постоянные, зависящие от производных высших порядков от решения.

Понятие согласованности (аппроксимации) разностной схемы означает, что для любого достаточно гладкого решения дифференциальной задачи u(x,t)

$$L_{h,\tau}u-Lu-f_h+f=\psi(h,\tau)\to 0, \quad \text{при} \quad h,\tau\to 0. \tag{2.5}$$

Величину $\psi(h,\tau)$ обычно называют погрешностью аппроксимации или невязкой разностной схемы. Обычно величина $\psi(h,\tau)$ находится из представления решения дифференциальной задачи в точках шаблона сетки в виде степенного ряда по степеням h и τ.

Эллиптические уравнения в прямоугольной области.

Многие актуальные приложения приводят к краевым задачам для уравнений в частных производных эллиптического типа, которые в трехмерной постановке имеют вид

$$\sum_{i,j=1}^{3} \frac{\partial}{\partial x_i} \lambda_{ij}(x) \frac{\partial u}{\partial x_j} = f(x), \quad x \in \Omega. \tag{2.6}$$

Примером эллиптического уравнения является уравнение Пуассона

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), \quad (x,y) \in \Omega = [0, L_x] \times [0, L_y], \tag{2.7}$$

с краевыми условиями Дирихле:

$$u(0,y) = u(L_x,y) = u(x,0) = u(x,L_y) = 0. \tag{2.8}$$

Пример 2.1. Рассмотрим задачи Дирихле для двумерного уравнения Пуассона

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} = 10 \sin(2\pi x/L) \sin(2\pi y/L), \\ -L/2 < x < L/2, \ -L/2 < y < L/2, \ L = 2. \end{cases}$$

с нулевыми и ненулевыми граничными условиями

$$u(\pm L/2, y) = u(x, \pm L/2) = 0.$$

$$u(\pm L/2, y) = y, \quad u(x, \pm L/2) = \pm L/2.$$

Программная реализация разностного метода для рассмотренных задач приведена в программном блоке, а результаты численного моделирования представлены на рис.2.1.

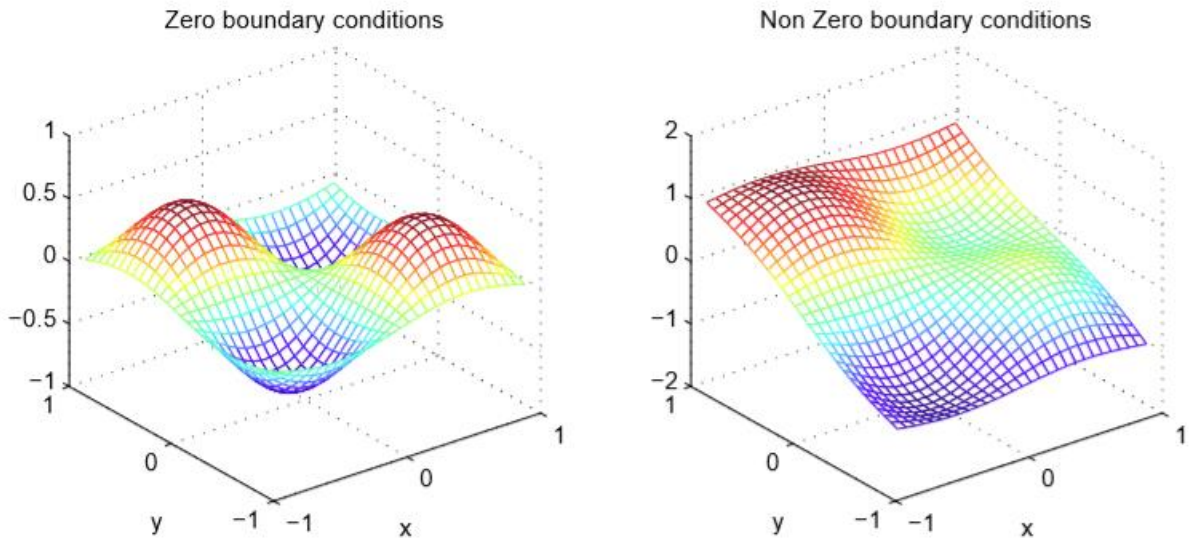


Рис.2.1. Решение двумерного уравнения Пуассона с нулевыми и ненулевыми условиями Дирихле

Нелинейные уравнения.

Область применимости линейных моделей всегда ограничена пределами входных данных, в рамках которых можно пренебречь зависимостью коэффициентов от решения и его производных. В силу этого, при моделировании прикладных инженерно-физических задач, адекватное описание реальности, как правило, достигается при использовании нелинейных уравнений в частных производных.

Нелинейные дифференциальные задачи весьма нетривиальны с точки зрения классического анализа, и численные методы во многих случаях выступают в качестве единственного универсального подхода, позволяющего получить по крайней мере приближенное решение. Более того, и с точки зрения численного анализа, нелинейные дифференциальные задачи во многих случаях также представляют немалую сложность.

Прежде всего, в отличие от линейных неявных разностных схем, дискретизация нелинейные уравнения в частных производных приводит к системам нелинейных уравнений, анализ которых сам по себе представляет серьезную проблему. Кроме того, при моделирование нелинейной динамики приходится иметь дело с описанием достаточно сложных режимов, таких как генерация ударных волн, диффузионный хаос, режимы с обострением и т.п. Как правило, большинство сложных нелинейных режимов характеризуется деградацией гладкости и уширением Фурье спектра решения, что неизбежно порождает проблемы с обеспечением согласованности и сходимости дискретных моделей. И, наконец, принципиальным моментом теории численных методов для нелинейных дифференциальных задач является то, что фундаментальное положение теоремы Лакса об эквивалентности согласованности и устойчивости с одной стороны и сходимости дискретной модели с другой стороны, не выполняется для случая нелинейных задач. Поэтому, если согласованность и устойчивость являются достаточными условиями сходимости в линейном случае, то для нелинейных задач это лишь необходимое условие сходимости.

Методы переменных направлений и дробных шагов.

Многие дифференциальные задачи могут быть сформулированы в виде операторных уравнений

$$\frac{\partial u}{\partial t} = Au, \tag{2.9}$$

с оператором A допускающим аддитивное представление

$$A = A_1 + A_2, \tag{2.10}$$

где компоненты A_1 и A_2 являются дифференциальными операторами, или их дискретными приближениями.

К обобщенной двухкомпонентной задаче, описанной (2.9) и (2.10), может применяться подход, получивший название метод переменных направлений

(МПН). В англоязычной литературе он известен под аббревиатурой ADI метод (alternative direction implicit method). МПН представляет собой одно из компромиссных решений, позволяющее совместить преимущество явных методов (простоту реализации) и неявных схем (безусловную устойчивость) в одной вычислительной технике. Для рассматриваемой задачи схема МПН имеет следующий вид:

$$\frac{U^{n+1/2}-U^{n}}{0.5\tau}=A_1U^{n+1/2}+A_2U^{n},$$

$$\frac{U^{n+1}-U^{n+1/2}}{0.5\tau}=A_1U^{n+1/2}+A_2U^{n+1}.$$

Каждое из приведенных выше уравнений представляют собой локально неявную схему по одному из пространственных направлений. Идея, положенная в основу двухкомпонентного МПН, состоит в чередовании двух альтернативных схем, каждая из которых явная по одному пространственному направлению и неявной по другому. Чередование этих схем приводит к удивительным результатам. Данный метод обладает безусловной устойчивости и его вычислительная сложность сравнима с методами решения одномерных задач, т.е. не выходит за пределы оптимальной вычислительной сложности. Кроме того, симметричность алгоритма на четном числе шагов позволяет получить второй порядок точности как по пространству, так и по времени.

Общая идея методов расщепления или методов дробных шагов. состоит в том, что эволюционный оператор задачи (2.9) представляется в виде суперпозиции более простых эволюционных операторов, простых в смысле вычислительной сложности. Другими словами, исходная задача разбивается на совокупность задач, которые решаются последовательно, циклически и решение каждой задачи выступают в качестве начальных условий для следующей задачи последовательности.

В отличие от МПН, метод расщепления может быть использован для произвольного числа компонент в аддитивном представлении оператора, включая случай неограниченных операторов. Схема МПН является безусловно устойчивой в случае двух компонент и с небольшими изменениями может быть обобщена на трехмерный случай. Метод расщепления наиболее эффективен при решении линейных и нелинейных задач, допускающих аддитивное представление оператора, в котором компоненты слабо связаны друг с другом.

Многосеточные методы.

Идея многосеточного метода состоит в отделении низкочастотных, медленно сходящихся составляющих решения с целью их независимой обработки на грубой сетке.

Пример 2.2. Продемонстрируем преимущество многосеточного метода на примере реализации разностной схемы для решения двумерного уравнения Пуассона:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \cos\left(\frac{\pi x}{L}\right)\cos\left(\frac{\pi y}{L}\right),$$

$$-L/2 < x < L/2, \quad -L/2 < y < L/2, \quad L = 2,$$

с нулевыми краевыми условиями

$$u(\pm L/2, y) = u(x, \pm L/2) = 0,$$

$$u(\pm L/2, y) = y, \quad u(x, \pm L/2) = x.$$

Аппроксимируем задачу разностной схемой. Далее, воспользуемся двухсеточным v-циклом, применяя МПИ для итераций сглаживания. В качестве операторов проекции и продолжения для перехода между грубой и мелкой сетками воспользуемся процедурами прореживания и интерполяции соответственно.

Программная реализация многосеточного алгоритма представлена в программном блоке, а некоторые иллюстрации результатов численных экспериментов представлены ниже. На рис. 2.2 показана динамика сходимости МПИ с оптимальным значением итерационного параметра при решении рассмотренной задачи на грубой и мелкой сетках. Как видно из представленных результатов, относительная погрешность 10^{-4} достигается на мелкой сетке примерно за 700 итераций, в то время как на грубой сетке число итераций не превосходит 200.

В случае двухсеточного метода с тремя итерациями сглаживания относительная погрешность 10−4 достигается после одного v-цикла независимо от размера сетки (см. рис. 2.3).

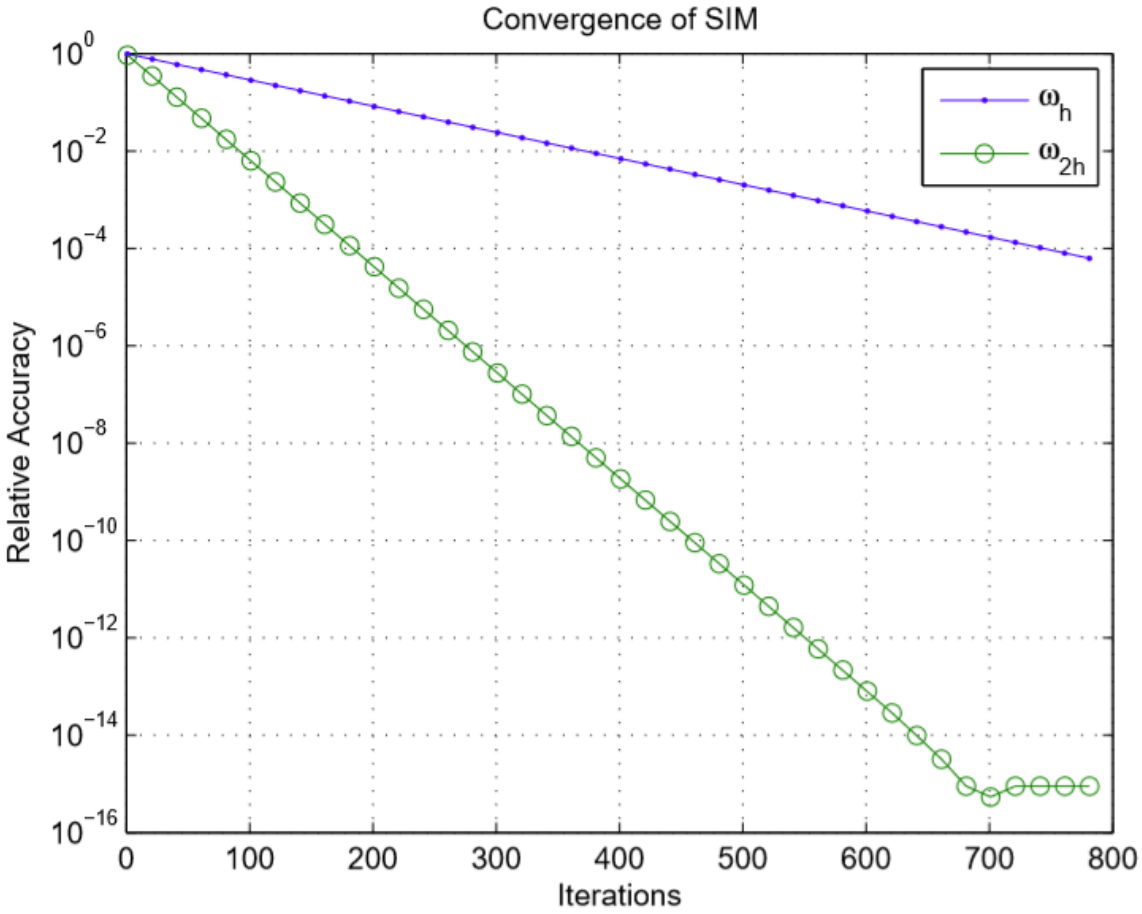


Рис.2.2. Сходимость МПИ на мелкой (ω_h) и грубой (ω_{2h}) сетках

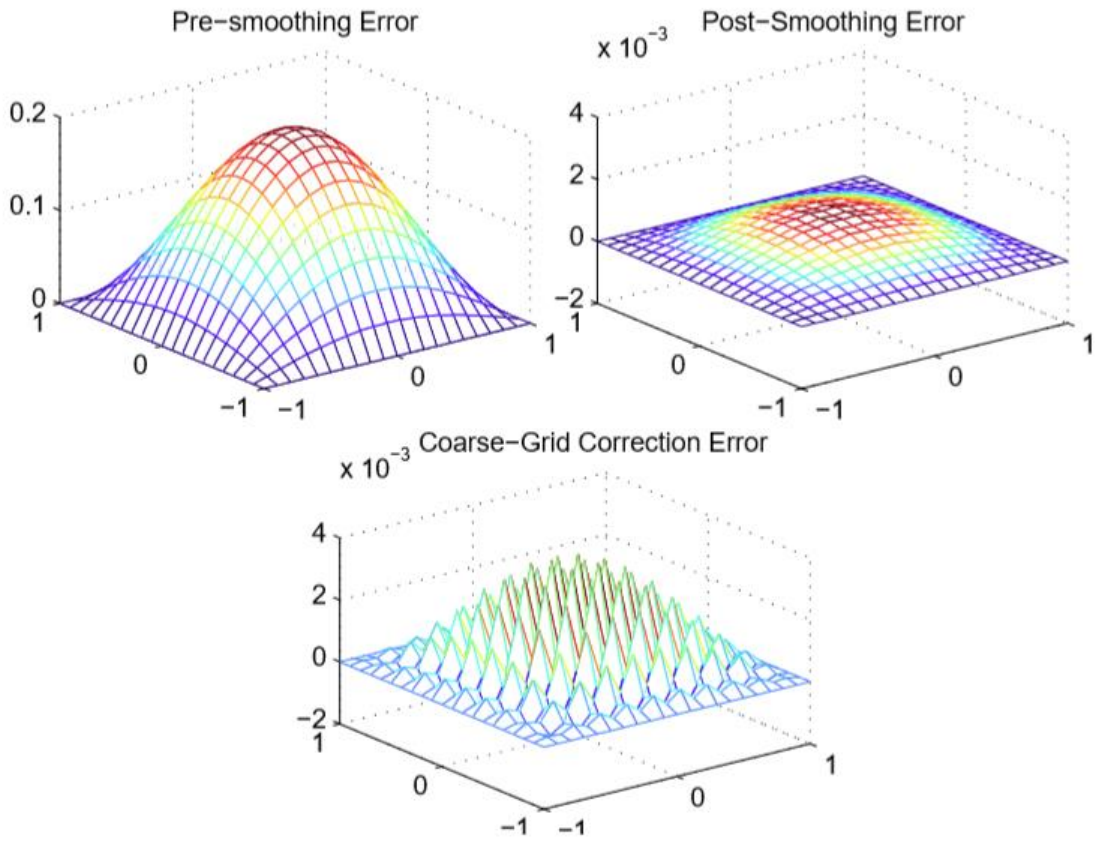


Рис.2.3. Динамика погрешности в процессе двухсеточного v-цикла

Тема 2.2. Метод конечных элементов

Слабая формулировка дифференциальной задачи.

Особенности метода конечных элементов рассмотрим применительно к решению задачи Дирихле для двумерного эллиптического уравнения в прямоугольной области Ω с границей $\partial\Omega$:

$$\frac{\partial}{\partial x}\lambda(x,y)\frac{\partial u}{\partial x}+\frac{\partial}{\partial y}\lambda(x,y)\frac{\partial u}{\partial y}=f(x,y),\quad (x,y)\in\Omega, \tag{2.11}$$

$$u(x,y)=0,\quad (x,y)\in\partial\Omega. \tag{2.12}$$

Первый шаг в построении метода конечных элементов состоит в переходе к слабой формулировке дифференциальной задачи. Слабая формулировка для (2.11) получается путем умножения данного уравнения на достаточно гладкую пробную функцию $v(x,y)$: $v(x,y) = 0$, $(x,y) \in \partial\Omega$ с последующим интегрированием полученного равенства по области Ω:

$$\iint\limits_{\Omega} \left[\frac{\partial}{\partial x} \lambda(x,y) \frac{\partial u}{\partial x} v + \frac{\partial}{\partial x} \lambda(x,y) \frac{\partial u}{\partial y} v \right] dxdy = \iint\limits_{\Omega} f(x,y) v \, dxdy. \tag{2.13}$$

Используя формулы интегрирования по частям будем иметь:

$$\iint\limits_{\Omega} \lambda(x,y) \left[\frac{\partial u}{\partial x} \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \frac{\partial v}{\partial y} \right] dxdy + \iint\limits_{\Omega} f(x,y) v \, dxdy = 0. \tag{2.14}$$

Уравнение (2.14) называют слабая формулировка краевой задачи (2.11) – (2.12), а u — слабое решение.

Предпочтительность слабой формулировки состоит в том, что интегральная форма снижает требования к дифференцируемости решения (требуется наличие только первых производных решения). Кроме того, слабая формулировка задачи упрощает дальнейшие шаги в построении конечно-элементной модели.

Конечно-элементная дискретизация.

Следующий шаг состоит в дискретизации области и выборе базисных функций. В двумерном случае область Ω разбивается обычно на треугольные подобласти. Однако, вообще говоря, для этих целей могут использоваться семейства произвольных многоугольников Ω_n (включая криволинейные многоугольники): $\overline{\Omega} = \cup \Omega_n$. Слабая формулировка задачи позволяет естественным образом провести декомпозицию области в систему подобластей, поскольку интегрирование в (2.14) может быть выполнено для каждой

подобласти Ω_n в отдельности. Следующая идея состоит в замене бесконечномерной задачи конечномерным аналогом. Полагаем, что решение задачи может быть представлено точно (или приближенно) бесконечной (или конечной) комбинацией некоторых базисных функций — функций формы. $\phi_k(x,y)$:

$$u(x,y) = \sum_{k=1}^{\infty} \alpha_k \psi_k(x,y) \simeq \sum_{k=1}^{N} \alpha_k \psi_k(x,y). \tag{2.15}$$

Приближенное решение и аппроксимирующее подпространство может быть получено путем перехода к конечномерному базису в представлении (2.15).

В качестве аппроксимирующего конечномерного подпространства $H^1{}_0$ будем использовать множество кусочно полиномиальных функций, удовлетворяющих краевым условиям и имеющий компактный носитель в пределах отдельной ячейки сетки. Простейший пример базисных функций — кусочно–линейные функции. Полагаем, что в двумерном случае аппроксимация выполняется в пределах треугольной ячейки сетки линейной функцией

Под конечными элементами понимают ячейку сетки с определенными на нем функции формы. В рассмотренном случае мы имеем дело с линейным треугольным конечным элементом. В случае квадратичных или кубических функций формы мы имеем дело с конечными элементами высшего порядка.

Смешанные производные и граничные условия.

Есть два класса стандартных граничных условий: краевые условия Дирихле и Неймана. В случае краевого условия Дирихле на границе области (или на некотором участке границы $\partial\Omega_1 \subset \partial\Omega$) задается явно значение искомого решения:

$$u(x,y) = u_g(x,y), \quad (x,y) \in \partial\Omega_1 \tag{2.16}$$

В случае краевых условий Неймана на границе области (или ее части $\partial\Omega_2 \in \partial\Omega$) определяется градиент искомого решения в виде:

$$\lambda \mathbf{n} \cdot \nabla u(x,y) = -qu + \omega, \quad (x,y) \in \partial\Omega_2, \tag{2.17}$$

где n — единичный вектор внешней нормали к границе области $\partial\Omega$.

Дискретная конечно-элементная модель.

После того как матрицы жесткости и векторы правых частей вычислены для каждого элемента в отдельности, следующим шагом является сборка дискретной модели. Данная модель представляет собой систему алгебраических уравнений, размерность которой совпадает с числом точек сетки.

Для простоты будем полагать, что все конечные элементы линейные и имеют однородную геометрию. Пример такой однородной триангуляции для случая прямоугольной области представлен на рис. 2.4.

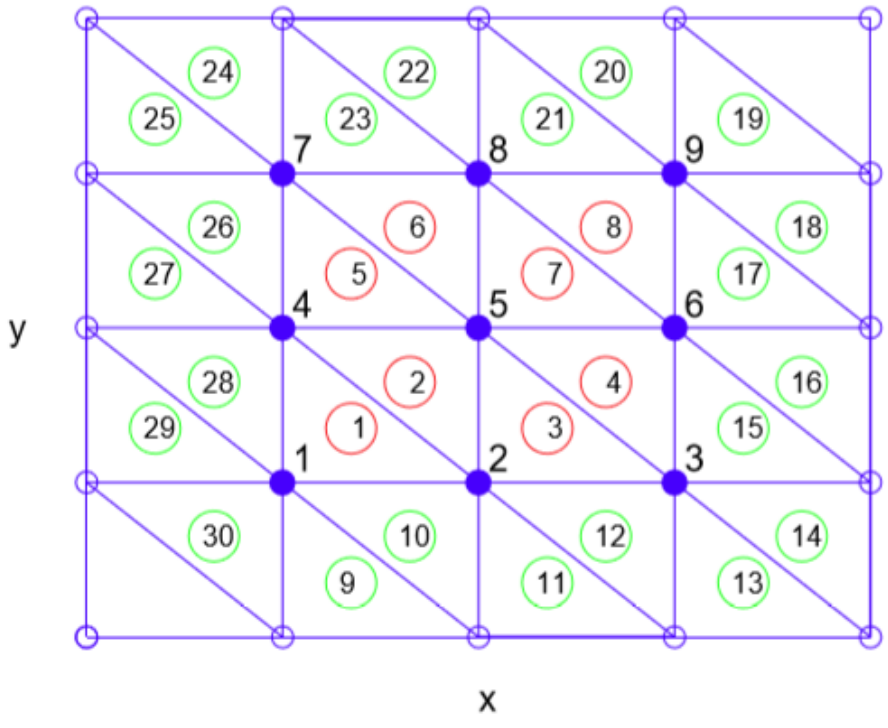


Рис.2.4. Однородная треугольная сетка в прямоугольной области

Вначале пронумеруем внутренние узлы сетки и внутренние треугольники. Для определенности используем нумерацию по строкам. Граничные элементы можно пронумеровать отдельно (см. рис. 2.4).

Процедура сборки не столь тривиальна, как может показаться на первый взгляд. Это связано с тем, что каждый узел сетки, за исключением угловых граничных точек, принадлежит одновременно нескольким соседним элементам. Например, каждый внутренний узел принадлежит одновременно шести элементам и вклад каждого должен быть адекватно учтен в итоговой дискретной модели.

Один из возможных алгоритмов сборки может быть представлен в виде двух стадий. На первом этапе мы собираем вместе все локальные матрицы жесткости в блок-диагональную черновую матрицу жесткости, согласно порядку нумерации элементов сетки. Размерность черновой матрицы жесткости определяется общим числом элементов и размерностью локальных матриц жесткости. Для нулевых краевых условий размерность локальной матрицы жесткости может быть понижена с учетом нулевого вклада граничных узлов. Следующий этап состоит в объединении строк черновой матрицы, которые связаны с одним и тем же узлом сетки.

Примеры прикладного программного обеспечения на основе метода конечных элементов.

Метод конечных элементов является одним из наиболее эффективных инструментов численного анализа уравнений с частными производными. В то же время данный метод является достаточно сложным и требует серьезной подготовки для реализации его возможностей применительно к тому многообразию дифференциальных задач, которые возникают в современных инженерно – физических приложениях. В связи с этим в инженерной практике разумно использовать профессионально разработанные программные средства, позволяющие упростить и сделать доступной данную технику для широкого

круга пользователей, не имеющих опыта разработки подобных приложений, но хорошо представляющих суть решаемых задач.

Среди программных продуктов, использующих метод конечных элементов, можно найти десятки интересны вариантом, начиная от простейших библиотек с отрытым кодом (Elmer, FreeFem++, OOFEM, Code Aster, OpenFOAM, Z88Aurora и т.п.) до продвинутых коммерческих проектов (COMSOL, ANSYS, SOLIDWORKS, Range Software и др.).

В качестве первого опыта работы с методом конечных элементов можно рекомендовать одно из наиболее простых, широко доступных и одновременно весьма эффективных приложений MATLAB Partial Differential Equations ToolboxTM (PDE app). Данное приложение включает набор функций, рассчитанных на решения основных типов дифференциальных уравнений (эллиптических параболических, гиперболических и проблему собственных функций) в двумерной постановке для области произвольной формы. Благодаря наличию весьма адекватного графического интерфейса работа с приложением достаточно удобна и доступна широкому кругу пользователей, причем глубокие знания и опыт программирования в среде MATLAB не являются обязательными условиями успешного использования приложения.

Раздел 3. Методы линейного программирования

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*
2. *Попова Н.В. Математические методы –- Электронный учебник, 2005 ВТК*
3. *А.Г.Трифонов. Постановка задачи оптимизации и численные методы ее решения.*
4. *ru.wikipedia.org*

Тема 3.1. Линейное программирование.

Симплекс-метод.

Под линейным программированием понимается раздел теории экстремальных задач, в котором изучаются задач и минимизации (или максимизации) линейных функций на множествах, задаваемых системами линейных равенств и неравенств. Каждое из условий-неравенств определяет полупространство, ограниченное гиперплоскостью. Пересечение полупространств образует выпуклый п-мерный многогранник Q. Условия равенства выделяют из n-мерного пространства (п-l) -мерную плоскость, пересечение которой с областью Q дает выпуклый (n-l) -мерный многогранник G. Экстремальное значение линейной формы (если оно существует) достигается в некоторой вершине многогранника. При вырождении оно может достигаться во всех точках ребра или грани многогранника.

Симплекс-метод является основным в линейном программировании. Решение задачи начинается с рассмотрений одной из вершин многогранника условий. Если исследуемая вершина не соответствует максимуму (минимуму), то переходят к соседней, увеличивая значение функции цели при решении задачи на максимум и уменьшая при решении задачи на минимум. Таким образом, переход от одной вершины к другой улучшает значение функции цели. Так как число вершин многогранника ограничено, то за конечное число шагов

гарантируется нахождение оптимального значения или установление того факта, что задача неразрешима.

Этот метод является универсальным, применимым к любой задаче линейного программирования в канонической форме. Система ограничений здесь - система линейных уравнений, в которой количество неизвестных больше количества уравнений. Если ранг системы равен ***r***, то мы можем выбрать ***r*** неизвестных, которые выразим через остальные неизвестные. Для определенности предположим, что выбраны первые, идущие подряд, неизвестные $X_1, X_2, \ldots, X_r$. Тогда наша система уравнений может быть записана как

$$\begin{cases} x_1 = b_1 + a_{1,r+1}x_{r+1} + \ldots + a_{1n}x_n, \\ x_2 = b_2 + a_{2,r+1}x_{r+1} + \ldots + a_{2n}x_n, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_r = b_r + a_{r,r+1}x_{r+1} + \ldots + a_{rn}x_n. \end{cases}$$

К такому виду можно привести любую совместную систему, например, методом Гаусса. Правда, не всегда можно выражать через остальные первые r неизвестных (мы это сделали для определенности записи). Однако такие r неизвестных обязательно найдутся. Эти неизвестные (переменные) называются базисными, остальные свободными.

Придавая определенные значения свободным переменным и вычисляя значения базисных (выраженных через свободные), мы будем получать различные решения нашей системы ограничений. Таким образом, можно получить любое ее решение. Нас будут интересовать особые решения, получаемые в случае, когда свободные переменные равны нулю. Такие решения называются базисными, их столько же, сколько различных базисных видов у данной системы ограничений. Базисное решение называется допустимым базисным решением или опорным решением, если в нем значения переменных

неотрицательны. Если в качестве базисных взяты переменные $x_1, x_2, ..., x_r$, то решение $\{b_1, b_2,..., b_r, 0, ..., 0\}$ будет опорным при условии, что $b_1, b_2,..., b_r \geq 0$.

Симплекс-метод представляет собой некоторую процедуру направленного перебора опорных решений. Исходя из некоторого, найденного заранее опорного решения по определенному алгоритму симплекс-метода мы подсчитываем новое опорное решение, на котором значение целевой функции F не меньше, чем на старом. После ряда шагов мы приходим к опорному решению, которое является оптимальным планом.

Итак, симплексный метод вносит определенный порядок как при нахождении первого (исходного) базисного решения, так и при переходе к другим базисным решениям. Его идея состоит в следующем.

Имея систему ограничений, приведенную к общему виду, то есть к системе m линейных уравнений с n переменными $(m < n)$, находят любое базисное решение этой системы, заботясь только о том, чтобы найти его как можно проще.

Если первое же найденное базисное решение оказалось допустимым, то проверяют его на оптимальность. Если оно не оптимально, то, осуществляется переход к другому, обязательно допустимому базисному решению.

Симплексный метод гарантирует, что при этом новом решении линейная форма, если и не достигнет оптимума, то приблизится к нему. С новым допустимым базисным решением поступают так же, пока не находят решение, которое является оптимальным.

Если первое найденное базисное решение окажется недопустимым, то с помощью симплексного метода осуществляется переход к другим базисным решениям, которые приближают нас к области допустимых решений, пока на каком-то шаге решения либо базисное решение окажется допустимым и к нему применяют алгоритм симплексного метода, либо мы убеждаемся в противоречивости системы ограничений.

Таким образом, применение симплексного метода распадается на два этапа: нахождение допустимого базисного решения системы ограничений или

установление факта ее несовместности; нахождение оптимального решения. При этом каждый этап может включать несколько шагов, соответствующих тому или иному базисному решению. Но так как число базисных решений всегда ограниченно, то ограниченно и число шагов симплексного метода.

Приведенная схема симплексного метода явно выражает его алгоритмический характер (характер четкого предписания о выполнении последовательных операций), что позволяет успешно программировать и реализовать этот метод на ЭВМ. Задачи же с небольшим числом переменных и ограничений могут быть решены симплексным методом вручную.

Двойственность.

В прямой задаче линейного программирования целевая функция является линейной комбинацией *n* переменных. Есть *m* ограничений, каждое из которых ограничивает сверху линейную комбинацию *n* переменных. Цель – максимизировать значение функции при ограничениях. Решение – это вектор из *n* значений, дающее максимальное значение целевой функции.

В двойственной задаче целевая функция является линейной комбинацией *m* значений, которые являются правыми частями *m* ограничений прямой задачи. Имеется *n* двойственных ограничений, каждое из которых ограничивает снизу линейную комбинацию *m* двойственных переменных.

Сопоставляя формы записи прямой и двойственной задач, можно установить между ними следующие взаимосвязи:

1. Если прямая задача является задачей максимизации, то двойственная будет задачей минимизации, и наоборот.
2. Коэффициенты целевой функции прямой задачи $c_1,...,c_n$ становятся свободными членами ограничений двойственной задачи.
3. Свободные члены ограничений прямой задачи $b_1,...,b_m$ становятся коэффициентами целевой функции двойственной задачи.
4. Матрица ограничений двойственной задачи получается путем транспонирования матрицы ограничений прямой задачи.

5. Знаки неравенств в ограничениях изменяются на противоположные.
6. Число ограничений прямой задачи равно числу переменных двойственной задачи, и наоборот.

Параметрическое линейное программирование.

Параметрическое программирование представляет собой один из разделов математического программирования, изучающий задачи, в которых целевая функция или ограничения зависят от одного или нескольких параметров.

Необходимость рассмотрения подобных задач обусловлена различными причинами. Одной из основных является та, что исходные данные для численного решения любой реальной задачи оптимизации в большинстве случаев определяются приближенно или могут изменяться под влиянием каких-то факторов, что может существенно сказаться на оптимальности выбираемой программы (плана) действий. Соответственно, разумно указывать не конкретные данные, а диапазон возможного изменения данных, чтобы в результате решения иметь наилучшие планы для любого варианта исходных данных.

С математической точки зрения параметрическое программирование выступает как одно из средств анализа чувствительности решения к вариации исходных данных, оценки устойчивости решения.

Существуют различные подходы к подобному анализу (например, на основе постановки двойственной задачи).

Модели линейного программирования.

Задачи, решаемые методами ЛП, очень разнообразны по содержанию. Но их математические модели схожи и условно объединяются в три большие группы задач:

- транспортные задачи;
- задачи о составлении плана;

- задачи о смеси (о диете).

Рассмотрим примеры конкретных экономических задач каждого типа, подробно остановимся на построении модели каждой задачи.

Транспортная задача

На двух торговых базах А и В имеется 30 гарнитуров мебели, по 15 на каждой. Всю мебель требуется доставить в два мебельных магазина, С и Д причем в С надо доставить 10 гарнитуров, а в Д - 20. Известно, что доставка одного гарнитура с базы А в магазин С обходится в одну денежную единицу, в магазин Д - в три денежных единицы. Соответственно с базы В в магазины С и Д: две и пять денежных единиц. Составить план перевозок так, чтобы стоимость всех перевозок была наименьшей.

Данные задачи для удобства разметим в таблице. На пересечении строк и столбцов стоят числа, характеризующие стоимость соответствующих перевозок (табл. 3.1).

Таблица 3.1

магазины / базы	С	Д	отправлено гарнитуров
А	1	3	15
В	2	5	15
получено гарнитуров	10	20	30 = 30

Составим математическую модель задачи.

Необходимо ввести переменные. В формулировке вопроса говорится, что необходимо составить план перевозок. Обозначим через x_1, x_2 количество гарнитуров, перевозимых с базы А в магазины С и Д соответственно, а через y_1, y_2 - количество гарнитуров, перевозимых с базы В в магазины С и Д

соответственно. Тогда количество мебели, вывозимое со склада А, равно (x_1 + x_2), а со склада В - (y_1 + y_2). Потребность магазина С равна 10 гарнитурам, и в него привезли (x_1 + y_1) штук, т. е. $x_1 + y_1 = 10$. Аналогично, для магазина Д имеем $x_2 + y_2 = 20$. Заметим, что потребности магазинов в точности равны количеству гарнитуров, имеющихся на складах, поэтому $x_1 + y_2 = 15$ и $y_1 + y_2 = 15$. Если бы со складов вы увезли меньше, чем по 15 комплектов, то магазинам не хватило бы мебели для удовлетворения их потребностей. Итак, переменные x_1, x_2, y_1, y_2 по смыслу задачи неотрицательны и удовлетворяют системе ограничений:

$$\begin{cases} x_1 + y_1 = 10; \\ x_2 + y_2 = 20; \\ x_1 + x_2 = 15; \\ y_1 + y_2 = 15; \\ x_i \geq 0, y_i \geq 0, i = 1,2. \end{cases} \quad (3.1)$$

Обозначив через F транспортные расходы, посчитаем их. на перевозку одного комплекта мебели из А в С тратится одна ден. ед., на перевозку x_1 комплектов - x_1 ден. ед. Аналогично, на перевозку x2 комплектов из А в Д затратится $3x_2$ ден. ед.; из В в С - $2y_1$ ден. ед., из В в Д - $5y_2$ ден. ед.

Итак,

$$F = 1x_1 + 3x_2 + 2y_1 + 5y_2 \rightarrow \min \quad (3.2)$$

(общая стоимость перевозок должна быть минимальной). Сформулируем задачу математически.

На множестве решений системы ограничений (3.1) найти такое решение, которое обращает в минимум целевую функцию F (3.2), или найти оптимальный план (x_1, x_2, y_1, y_2), определяемый системой ограничений (3.1) и целевой функцией (3.2).

Перейти к онлайн решению своей задачи

Задача о составлении плана

Некоторому заводу требуется составить оптимальный план выпуска двух видов изделий, которые обрабатываются на четырех видах машин. Известны определенные возможности и производительность оборудования; цена изделий, обеспечивающая прибыль заводу, составляет 4 тыс. руб. за изделие I вида, 6 тыс. руб. - за изделие II вида. Составить план выпуска этих изделий так, чтобы от реализации их завод получил наибольшую прибыль. В таблице указано время, необходимое для обработки каждого из двух видов изделий на оборудовании всех четырех видов (табл. 3.2).

Таблица 3.2

Изделия	Виды машин			
	1	2	3	4
I	1	0,5	1	0
II	1	1	0	1
Возможное время работы машин	18	12	12	9

Построим математическую модель.

В задаче необходимо определить план выпуска изделий, обозначим за x количество изделий I вида, за y - количество изделий II вида. Тогда посчитаем, сколько времени затратит первая машина на обработку всех производственных изделий. Она тратит одну единицу времени на одного изделие I вида, значит на x штук изделий потратит 1x ед. времени, на обработку y изделий II вида затратится 1y ед. времени. Всего резерв времени работы первой машины - 18 единиц времени. Значит, $x + y \leq 18$. Аналогичные рассуждения со второй машиной, третьей и четвертой дадут систему ограничений:

$$\begin{cases} x+y \le 18; \\ 0{,}5x+y \le 12; \\ x \le 12; \\ y \le 9; \\ x \ge 0, y \ge 0. \end{cases} \quad (3.3)$$

Общая прибыль будет выражена в целевой функции:

$$F = 4x + 6y \rightarrow \max. \quad (3.4)$$

Задача состоит в нахождении на множестве решений системы (3.3) такого решения, при котором значение целевой функции (3.4) было бы максимальным.

Задача составления смеси

Еще одна распространенная задача ЛП - задача о составлении смеси. Примером таких задач может быть задача о составлении таких смесей нефтепродуктов, которые бы удовлетворяли определенным техническим требованиям и были наиболее дешевыми по стоимости. Либо задачи о рационе, когда известна потребность в определенных веществах и содержание этих веществ в различных продуктах. Необходимо составить рацион так, чтобы удовлетворить потребности в необходимых веществах, и при этом продуктовая корзина имела бы минимальную стоимость при заданных ценах на продукты. Практически подобные задачи ставятся, к примеру, в любом животноводческом хозяйстве и имеют очень большой спектр применения.

Рассмотрим пример. Для откорма цыплят на птицефабрике в их рацион необходимо включать не менее 33 единиц вещества А, 23 единиц питательного вещества В, 12 единиц С. Для откорма используются три вида корма. Данные о содержании питательных веществ в каждом виде корма заданы таблицей. Также известна стоимость кормов. Необходимо составить наиболее дешевый рацион (табл. 3.3).

Таблица 3.3

Корма-продукты	Вещества			Стоимость 1 ед. корма
	A	B	C	
I	4	3	1	20
II	3	2	1	20
III	2	1	2	10

Для понимания задачи можете представить себе, что вещества A, B, C - это жиры, белки, углеводы, а продукты I, II, III - то, чем кормят цыплят, например, пшено, комбикорм, витаминные добавки. Тогда первая строка таблицы показывает содержание в одной единице пшена: 4 ед. белка, 3 ед. жиров, одной ед. углеводов. Вторая строка - содержание белков, жиров, углеводов в 1 ед. II продукта и т. д.

Если постановка задачи ясна, приступим к построению математической модели. В качестве ответа на поставленную задачу мы должны предложить рацион, т. е. указать сколько и каких кормов взять, чтобы необходимое количество питательных веществ было соблюдено и при этом он стоил как можно дешевле.

Поэтому, обозначим за x_1 количество кормов типа I в рационе, за x_2 - количество кормов типа II и, соответственно, x_3 - количество корма III в рационе. Тогда, вещества A при употреблении такого рациона цыплята получат $4x_1$ - при потреблении продуктов типа I, $3x_2$ - при потреблении II продукта, $2x_3$ - при потреблении III. Всего вещества A необходимо употребить по условию задачи не менее 33 единиц, следовательно, $4x_1 + 3x_2 + 2x_3 \geq 33$.

Аналогично рассуждая с веществами B и C, имеем: $3x_1 + 2x_2 + 1x_3 \geq 23$ и $x_1 + x_2 + 2x_3 \geq 12$.

Таким образом, получим систему ограничений:

$$\begin{cases} 4x_1 + 3x_2 + 2x_3 \geq 33; \\ 3x_1 + 2x_2 + 1x_3 \geq 23; \\ x_1 + x_2 + 2x_3 \geq 12; \\ x_1 \geq 0; \\ x_2 \geq 0; \\ x_3 \geq 0. \end{cases} \quad (3.5)$$

Переменные неотрицательны по смыслу задачи. При этом стоимость рациона выражается функцией:

$$F = 20x_1 + 20x_2 + 10x_3 \rightarrow \min, \quad (3.6)$$

т. к. 20, 20, 10 - стоимость одной ед. продуктов I, II, III типов соответственно, а в рационе их содержится x1, x2, x3 единиц.

Система ограничений (3.5) вместе с целевой функцией (3.6) и составляют математическую модель исходной задачи. Решить ее - значит найти x1, x2, x3, удовлетворяющие системе ограничений и обращающие значение функции F в минимальное.

Тема 3.2. Целочисленное линейное программирование

Введение.

Под задачей целочисленного линейного программирования (ЦЛП) понимается задача линейного программирования, в которой некоторые (а возможно, и все) переменные должны принимать целые значения. Задача ЦЛП называется полностью целочисленной, если все её переменные должны быть целочисленными. Для смешанной задачи ЦЛП лишь некоторые переменные предполагаются целочисленными, а остальные могут принимать произвольные (нецелые) значения.

Примеры задач целочисленного программирования.

Целочисленные задачи математического программирования могут возникать различными путями.

1. Существуют задачи линейного программирования, которые формально к целочисленным не относятся, но при соответствующих исходных данных всегда обладают целочисленным планом. Примеры таких задач – транспортная задача и ее модификации (задачи о назначениях, о потоках в сетях).

2. Толчком к изучению целочисленных задач в собственном смысле слова явилось рассмотрение задач линейного программирования, в которых переменные представляли физически неделимые величины. Они были названы задачами с неделимостью. Таковы, например, задачи об оптимизации комплекса средств доставки грузов, о нахождении минимального порожнего пробега автомобилей при выполнении заданного плана перевозок, об определении оптимального машинного парка и его оптимального распределения по указанным работам при условии минимизации суммарной стоимости (машинного парка и производимых работ), о нахождении минимального количества судов для осуществления данного графика перевозок и т. п.

3. Другим важным толчком к построению теории целочисленного программирования стал новый подход к некоторым экстремальным комбинаторным задачам. В них требуется найти экстремум целочисленной линейной функции, заданной на конечном множестве элементов. Такие задачи принято называть задачами с альтернативными переменными. В качестве примеров можно назвать задачи коммивояжера (бродячего торговца), об оптимальном назначении, теории расписания, или календарного планирования, и задачи с дополнительными логическими условиями (например, типа «или – или», «если – то» и т. п.).

Исторически первой задачей целочисленного типа является опубликованная в 1932 г. венгерским математиком Е. Эгервари задача о назначении персонала. В 1955 г. на Втором симпозиуме по линейному

программированию была рассмотрена задача о бомбардировщике, известная как задача о ранце.

Методы решения задач целочисленного программирования.

В общем же случае для решения задач ЦЛП требуются специальные методы. В настоящее время существует несколько таких методов, из которых наиболее известным является метод Гомори, в основе которого лежит описанный выше симплексный метод.

Процесс определения оптимального плана задачи целочисленного программирования методом Гомори включает следующие основные этапы:

1. Используя симплексный метод, находят решение задачи без учета требования целочисленности переменных.

2. Составляют дополнительное ограничение для переменной, которая в оптимальном плане задачи имеет максимальное дробное значение, а в оптимальном плане задачи должна быть целочисленной.

3. Используя двойственный симплекс–метод, находят решение задачи, получающейся из исходной задачи в результате присоединения дополнительного ограничения.

4. В случае необходимости составляют еще одно дополнительное ограничение и продолжают итерационный процесс до получения оптимального плана исходной задачи или установления ее неразрешимости.

Тема 3.3. Методы решения транспортных задач

Постановка задачи и стратегии решения. Методы нахождения начального плана перевозок. Метод потенциалов

Транспортная задача (задача Монжа - Канторовича) - математическая задача линейного программирования специального вида о поиске оптимального распределения однородных объектов из аккумулятора к приемникам с минимизацией затрат на перемещение.

Для простоты понимания рассматривается как задача об оптимальном плане перевозок грузов из пунктов отправления (например, складов) в пункты потребления (например, магазины), с минимальными общими затратами на перевозки.

Математическая модель транспортной задачи описана выше.

Решить транспортную задачу можно различными методами, начиная от симплекс-метода и простого перебора, и заканчивая методом графов. Один из наиболее применяемых и подходящих для большинства случаев методов – итерационное улучшение плана перевозок.

Суть его в следующем: находим некий опорный план и проверяем его на оптимальность ($F \rightarrow \min$). Если план оптимален – решение найдено. Если нет – улучшает план столько раз, сколько потребуется, пока не будет найден оптимальный план.

Ниже приведен алгоритм решения транспортной задачи в самом общем виде:

1. Построение транспортной таблицы.
2. Проверка задачи на закрытость.
3. Составление опорного плана.
4. Проверка опорного плана на вырожденность.
5. Вычисление потенциалов для плана перевозки.
6. Проверка опорного плана на оптимальность.
7. Перераспределение поставок.
8. Если оптимальное решение найдено, переходим к п. 9, если нет – к п. 5.
9. Вычисление общих затрат на перевозку груза.
10. Построение графа перевозок.

1. Построение транспортной таблицы

Строим таблицу, где указываем запасы материалов, имеющиеся на складах поставщиков (A_i), и потребности заводов (B_j) в этих материалах.

В нижний правый угол ячеек таблицы заносим значение тарифов на перевозку груза (C_{ij}).

Магазины / Склады	B1	B2	B3	Запасы
A1	5	3	1	10
A2	3	2	4	20
A3	4	1	2	30
Потребности	15	20	25	

2. Проверка задачи на закрытость

Обозначим суммарный запас груза у всех поставщиков символом A, а суммарную потребность в грузе у всех потребителей – символом B.

Тогда:

$$A = \sum_{i=1}^{m} A_i \qquad B = \sum_{j=1}^{n} B_j$$

Транспортная задача называется **закрытой**, если A = B . Если же A ≠ B , то транспортная задача называется **открытой**. В случае закрытой задачи от поставщиков будут вывезены все запасы груза, и все заявки потребителей будут удовлетворены. В случае открытой задачи для ее решения придется вводить фиктивных поставщиков или потребителей.

Проверим задачу на закрытость:

A = 10 + 20 + 30 = 60

B = 15 + 20 + 25 = 60

A = B, следовательно, данная транспортная задача – закрытая.

3. Составление опорного плана

Составляет предварительный (опорный) план перевозок. Он не обязательно должен быть оптимальный. Это просто своеобразный «черновик», «набросок», улучшая который мы постепенно придем к плану оптимальному.

Есть разные методы нахождения опорного плана. Наиболее распространены следующие:

а) Метод Северо-Западного угла.

Суть метода проста - ячейки транспортной таблицы последовательно заполняются максимально возможными объемами перевозок, в направлении сверху вниз и слева направо. То есть сперва заполняется самая верхняя левая ячейка ("северо-западная" ячейка), потом следующая справа и т.д. Затем переходят на новую строку и вновь заполняют ее слева направо. И так пока таблица не будет заполнена полностью.

б) Метод минимального элемента.

Метод заключается в том, что для заполнения ячеек транспортной таблицы выбирается клетка с минимальным значением тарифа. Затем выбирается следующая клетка с наименьшим тарифом и так продолжается до тех пор, пока таблица не будет заполнена (все запасы и потребности при этом обнулятся).

в) Аппроксимация Фогеля.

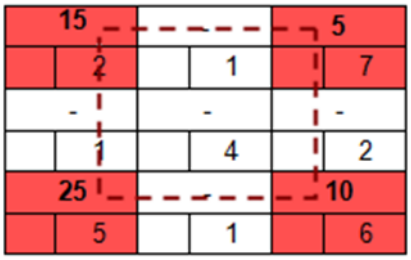
Основа метода в нахождении разности (по модулю) между парой минимальных тарифов в каждой строке и столбце. Затем в строке или столбце с наибольшей разностью заполняется клетка с наименьшим тарифом. Затем все эти действия повторяются заново, только при этом уже не учитываются заполненные клетки.

г*) Метод двойного предпочтения.*

Суть метода в том, что отмечаются клетки с наименьшим тарифом по строкам, а затем по столбцам. Затем ячейки заполняются в следующей очередности: сначала клетки с двумя отметками, потом с одной, наконец без отметок.

4. Проверка опорного плана на вырожденность

Клетки таблицы, в которые записаны отличные от нуля перевозки, называются базисными, а остальные (пустые) - свободными. План называется вырожденным, если количество базисных клеток в нем меньше, чем m + n -1. Если во время решения задачи получился вырожденный план, то его необходимо пополнить, проставив в недостающем числе клеток нулевую перевозку и превратив, тем самым, эти клетки в базисные (общий баланс и суммарная стоимость перевозок плана при этом не изменятся). Однако проводить пополнение плана, выбирая клетки произвольно, нельзя. План должен быть ациклическим! План называется ациклическим, если его базисные клетки не содержат циклов. Циклом в транспортной таблице называется несколько клеток, соединенных замкнутой ломаной линией так, чтобы две соседние вершины ломаной были расположены либо в одной строке, либо в одном столбце. Ниже приведен пример цикла:



Ломаная линия может иметь точки самопересечения, но не в клетках цикла.

Кол-во базисных клеток = 5

m + n – 1 = 3 + 3 – 1 = 5

Следовательно, первоначальный план перевозок – невырожденный.

5. Вычисление потенциалов для плана перевозки

Для анализа полученных планов и их последующего улучшения удобно ввести дополнительные характеристики пунктов отправления и назначения, называемые потенциалами. Этот метод улучшения плана перевозок называется

методом потенциалов. Есть другие методы итерационного улучшения плана перевозок, но здесь мы их рассматривать не будем.

Итак, сопоставим каждому поставщику A_i и каждому потребителю B_j величины U_i и V_j соответственно так, чтобы для всех базисных клеток плана было выполнено соотношение:

$U_i + V_j = C_{ij}$

Добавим к транспортной таблице дополнительную строку и столбец для U_i и V_j.

Магазины / Склады	B1	B2	B3	U
A1	- 5	- 3	10 1	
A2	15 3	- 2	5 4	
A3	- 4	20 1	10 2	
V				

Предположим, что U1 = 0.

Магазины / Склады	B1	B2	B3	U
A1	- 5	- 3	10 1	0
A2	15 3	- 2	5 4	
A3	- 4	20 1	10 2	
V				

Тогда мы сможем найти V3 = C13 – U1 = 1 – 0 = 1.

Магазины / Склады	B1	B2	B3	U
A1	-	-	10	0
	5	3	1	
A2	15	-	5	
	3	2	4	
A3	-	20	10	
	4	1	2	
V			1	

Зная V3, мы теперь можем найти U3:

Магазины / Склады	B1	B2	B3	U
A1	-	-	10	0
	5	3	1	
A2	15	-	5	
	3	2	4	
A3	-	20	10	1
	4	1	2	
V			1	

По аналогии вычисляем все оставшиеся потенциалы:

Магазины / Склады	B1	B2	B3	U
A1	-	-	10	0
	5	3	1	
A2	15	-	5	3
	3	2	4	
A3	-	20	10	1
	4	1	2	
V	0	0	1	

6. Проверка плана на оптимальность методом потенциалов

Для каждой свободной клетки плана вычислим разности

$\Delta C_{ij} = C_{ij} – (U_i + V_j)$

и запишем полученные значения в левых нижних углах соответствующих ячеек.

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		0
	5	5	*3*	3	*0*	1	
A2	15		-		5		3
	0	3	*-1*	2	*0*	4	
A3	-		20		10		1
	3	4	*0*	1	*0*	2	
V	0		0		1		

План является оптимальным, если все разности $\Delta C_{ij} \geq 0$.

В данном случае план – неоптимальный ($\Delta C22 < 0$), и его следует улучшить путем перераспределения поставок.

7. Перераспределение поставок

Найдем ячейку с наибольшей по абсолютной величине отрицательной разностью ΔC_{ij} и построим цикл, в котором кроме этой клетки все остальные являются базисными. Такой цикл всегда существует и единственен.

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		0
	5	5	*3*	3	*0*	1	
A2	15		-		5		3
	0	3	*-1*	2	*0*	4	
A3	-		20		10		1
	3	4	*0*	1	*0*	2	
V	0		0		1		

Отметим ячейку с отрицательной разностью ΔC_{ij} знаком «+», следующую знаком «-», и так далее, поочередно.

Затем находим минимальной значение груза в ячейках цикла имеющих знак «-» (здесь это 5) и вписываем его в свободную ячейку со знаком «+». Затем последовательно обходим все ячейки цикла, поочередно вычитая и прибавляя к ним минимальное значение (в соответствии со знаками, которыми эти ячейки помечены: где минус - вычитаем, где плюс - прибавляем).

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		0
	5	5	*3*	3	*0*	1	
A2	15		+5		5-5		3
	0	3	*-1*	2	*0*	4	
A3	-		20-5		10+5		1
	3	4	*0*	1	*0*	2	
V	0		0		1		

Получим новый опорный план перевозок:

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		
		5		3		1	
A2	15		5		0		
		3		2		4	
A3	-		15		15		
		4		1		2	
V							

Так как базисных клеток стало больше, чем m + n – 1, то базисную клетку с нулевым значением делаем свободной:

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		
		5		3		1	
A2	15		5		-		
		3		2		4	
A3	-		15		15		
		4		1		2	
V							

Снова вычисляем значения потенциалов и разности ΔC_{ij}:

Магазины / Склады	B1		B2		B3		U
A1	-		-		10		0
	4	5	*3*	3	*0*	1	
A2	15		5		-		2
	0	3	*0*	2	*1*	4	
A3	-		15		15		1
	2	4	*0*	1	*0*	2	
V	1		0		1		

На этот раз все разности ΔC_{ij} ячеек положительные, следовательно, найдено оптимальное решение.

8. Если оптимальное решение найдено, переходим к п. 9, если нет – к п. 5.

У нас оптимальное решение найдено, поэтому переходим к пункту 9.

9. Вычисление общих затрат на перевозку груза

Вычислим общие затраты на перевозку груза (Z), соответствующие найденному нами оптимальному плану, по формуле:

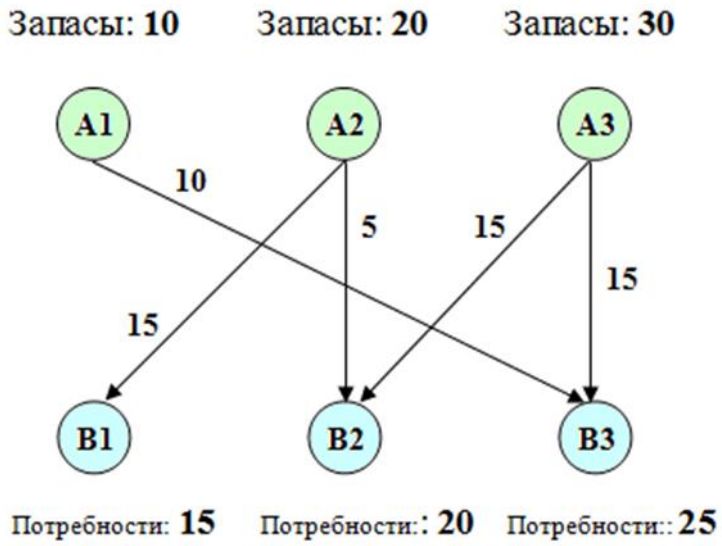
$$Fmin = 10 \cdot 1 + 15 \cdot 3 + 5 \cdot 2 + 15 \cdot 1 + 15 \cdot 2 = 110 \text{ ден. ед.}$$

Общие затраты на доставку всей продукции, для оптимального решения, составляют 110 ден. ед.

10. Построение графа перевозок

Найдя оптимальный план перевозок, построим граф. Вершинами графа будут «склады» и «магазины». В вершинах укажем соответствующие объемы запасов и потребностей. Дугам, соединяющим вершины графа, будут соответствовать ненулевые перевозки. Каждую такую дугу подпишем, указав объем перевозимого груза.

В результате получится граф, аналогичный изображенному ниже:



Раздел 4. Численные методы поиска безусловного экстремума

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*

Тема 4.1. Основные методы безусловной оптимизации

Методы нулевого порядка.

В типичных методах нулевого порядка направление поиска полностью определяется на основании последовательных вычислений целевой функции $f(\boldsymbol{x})$. Эти методы не требуют регулярности и непрерывности целевой функции и существования производных. Это является существенным достоинством при решении сложных прикладных задач.

Рассмотрим кратко стратегии поиска, применяемые в наиболее известных методах нулевого порядка.

Метод покоординатного спуска (метод Гаусса)

Метод покоординатного спуска (метод Гаусса) на каждой итерации по очереди минимизирует целевую функцию вдоль каждой из координат. На каждой итерации решение улучшается за счет минимизации вдоль выбранной i-ой координаты (компоненты) x_i вектора $x \in D$ при фиксированных остальных координатах.

Результаты работы метода Гаусса с использованием метода деления пополам в качестве метода линейного поиска приведены на Рис. 4.1, скрипт Matlab приведен в программном блоке.

Метод не всегда дает точный результат. Погрешности могут возникнуть как из-за погрешностей в линейном поиске, так и из-за особенностей целевой функции.

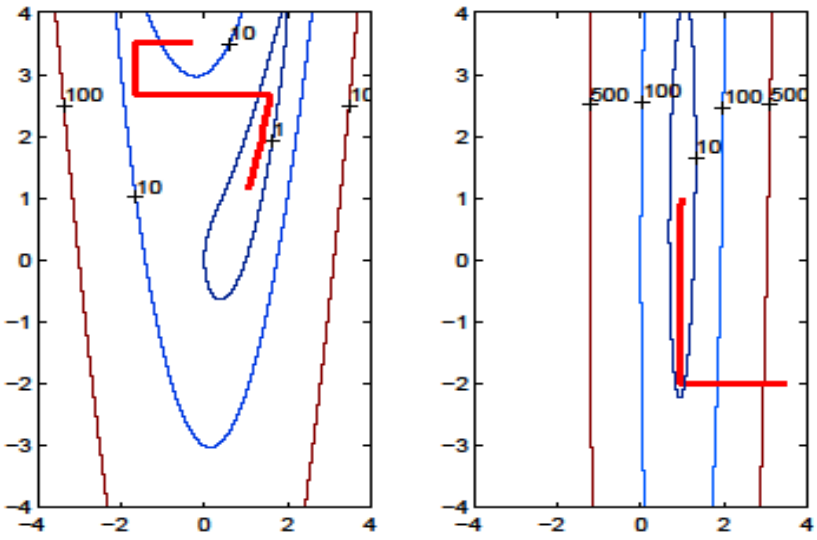


Рис. 4.1 – Результаты работы метода Гаусса

Метод Хука–Дживса (метод конфигураций)

Метод Хука–Дживса заключается в комбинации двух шагов: исследующего шага и шага по образцу. С помощью исследующего шага определяется предпочтительное направление для дальнейшего поиска по образцу. Величина исследующего шага вдоль каждой координаты может быть различной и может меняться в процессе исследующего шага. Возможны модификации алгоритма, в которых в процессе исследующего шага выполняется минимизация целевой функции по каждой координате с помощью какого-нибудь линейного поиска. Кроме того, в процессе шага по образцу можно также искать минимум функции с помощью какого-нибудь линейного поиска или изменять шаг в зависимости от значения функции $f(x)$.

Результаты работы метода Хука–Дживса с использованием минимизации функции (деление пополам) на исследующем шаге и на этапе поиска по образцу показаны на Рис. 4.2 и 4.3.

Так же, как и в методе Гаусса, в случае сильно вытянутых, изогнутых и круто изгибающихся линий уровня, метод Хука–Дживса может работать со значительными погрешностями.

Метод Розенброка

Метод Розенброка является итерационной процедурой, в нем, подобно методу Хука–Дживса, предусмотрен исследующий поиск. Однако, если в методе Хука – Дживса направления исследующего поиска фиксированы, то в методе Розенброка выбор этих направлений проводят на основании анализа скорости изменения целевой функции.

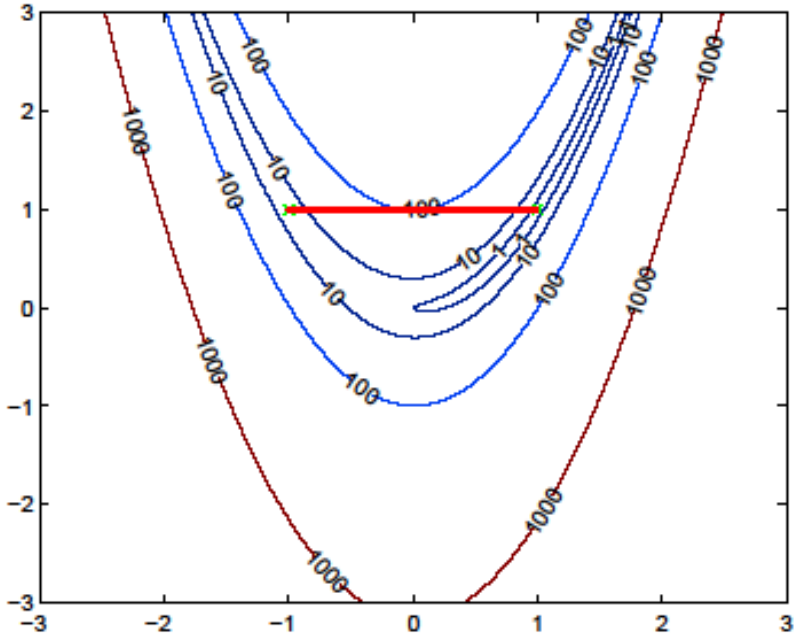


Рис. 4.2. Результат работы метода Хука–Дживса с использованием метода деления пополам на исследующем шаге

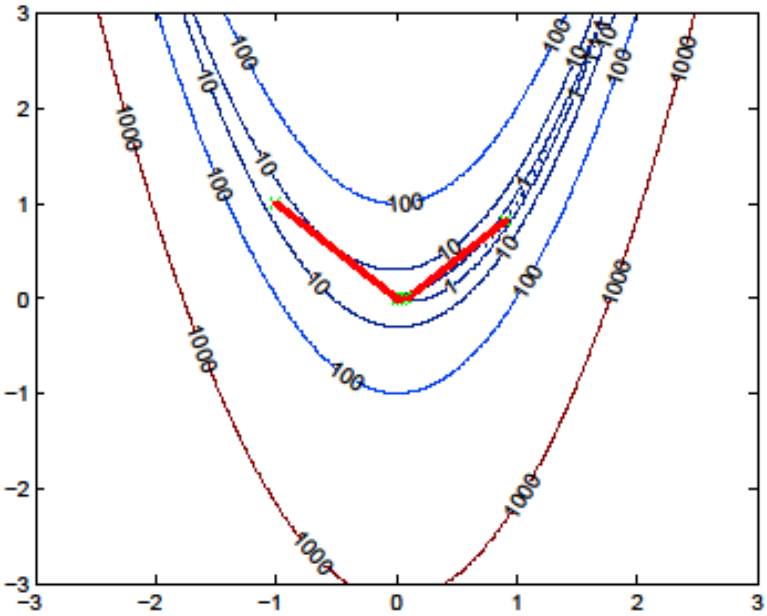


Рис. 4.3. Результаты работы метода Хука–Дживса с использованием метода деления пополам на этапе поиска по образцу

Выбор новых направлений координатных осей d^k на каждой *k*–ой итерации определяется с помощью процедуры Gram-Smidt:

$$a_j = \begin{cases} \mathbf{d}_j, & \text{if} \quad s_j = 0, \\ \sum_{i=j}^{N} \mathbf{d}_i s_i, & \text{if} \quad s_j \neq 0, \end{cases} \qquad b_j = \begin{cases} a_j, & \text{if} \quad j = 1, \\ a_j - \sum_{i=1}^{j-1} (a_i, \mathbf{d}_i)\mathbf{d}_i, & \text{if} \quad j \geqslant 2, \end{cases}$$

где $\mathbf{d}_j = b_j \|b_j\|^{-1}$.

Этот метод хорошо приспособлен для поиска оптимума целевой функции, которая представляет собой узкий овраг (или гребень), произвольно расположенный по отношению к варьируемым переменным.

Результаты работы метода Розенброка приведены ниже на Рис. 4.4. Скрипт Matlab приведен в программном блоке.

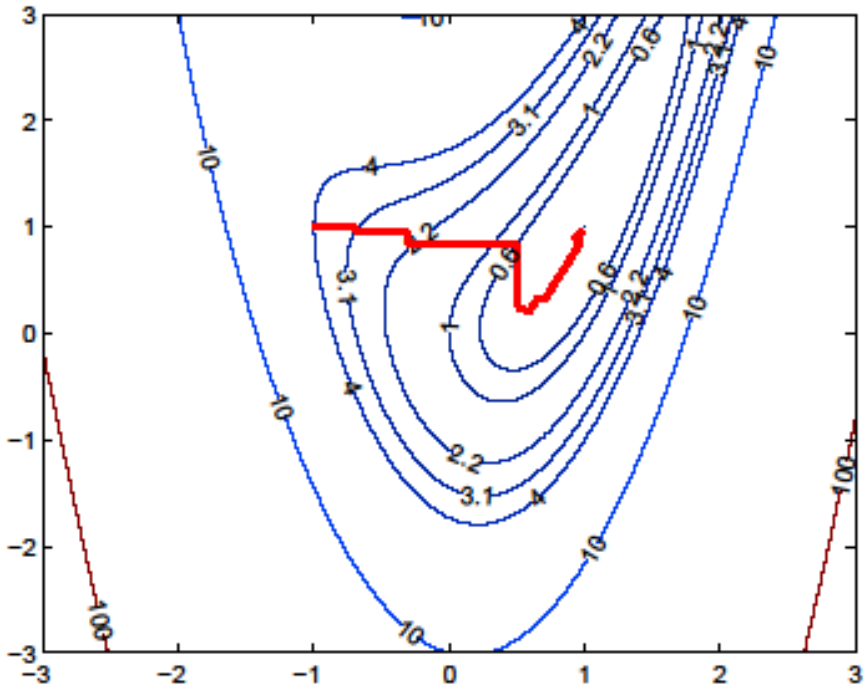


Рис. 4.4. Результат работы метода Розенброка

Методы случайного поиска

Методы случайного поиска относятся к наиболее простым методам поиска оптимума целевой функции. Однако при этом они могут быть весьма эффективны для минимизации различных функций. Случайность в этих методах чаще всего связана с направлением поиска, которое определяется с помощью случайного вектора единичной длины ξ^k.

Существуют различные варианты методов случайного поиска. Рассмотрим некоторые из них.

Случайный поиск с постоянным шагом. В алгоритме с возвратом при неудачном шаге (в случае, если $f(y) > f(x^k)$) происходит возврат в текущую точку x^k и поиск продолжается. Если число неудачных шагов из текущей точки достигнет некоторого числа M, то алгоритм останавливается, и x^k является искомым приближением.

В алгоритме наилучшей пробы на текущей итерации при помощи генерирования случайных векторов ξ^k получается M точек, лежащих на гиперсфере радиуса s^k с центром в точке x^k. Среди полученных точек выбирается такая точка y, в которой значение функции наименьшее. Если в выбранной точке значение функции меньше, чем в центре, т.е. $f(y) < f(x^k)$, то дальнейший поиск продолжается из этой точки. Иначе алгоритм останавливается, и точка x^k является искомым приближением.

Модифицированный алгоритм наилучшей пробы. Аналогично предыдущему алгоритму определяется точка y с наилучшим значением функции. Затем проводят поиск вдоль прямой, проходящей через x^k и y, и определяют точку x^{k+1}, в которой функция $f(x)$ будет иметь свое минимальное значение.

Методы первого порядка.

Методы первого порядка для поиска оптимума функции используют градиент этой функции. Градиент - это вектор в точке x, направление которого указывает направление (локальное) наибольшего возрастания функции *f*. Направление, противоположное направлению градиента, является направлением для минимизации или направлением наискорейшего спуска:

$$d^k = -\nabla f(x^k).$$

Тогда алгоритмы поиска оптимальной точки x^* с минимальным значением функции могут быть построены на основании информации, по крайней мере, о градиенте этой функции. Такие методы называются градиентными.

Градиентные методы безусловной оптимизации различаются способом построения последовательности точек x^k и способом выбора размера шага s^k.

Рассмотрим кратко особенности стратегий поиска, применяемых в наиболее известных методах первого порядка.

Градиентный спуск с постоянным шагом

В этом методе величина шага s^k задается пользователем и остается постоянной до тех пор, пока функция убывает в точках последовательности x^k. На каждой итерации метод «приближается» к решению, вычисляя новое значение x^k в соответствии с формулой:

$$x^{k+1} = x^k - s^k \nabla f(x^k).$$

Выбрать подходящий размер шага достаточно сложно. Как видно из Рис. 4.5, слишком маленький шаг s^k может вызвать очень медленную сходимость алгоритма. С другой стороны, слишком большой шаг s^k может привести к тому, что алгоритм не отыщет минимум.

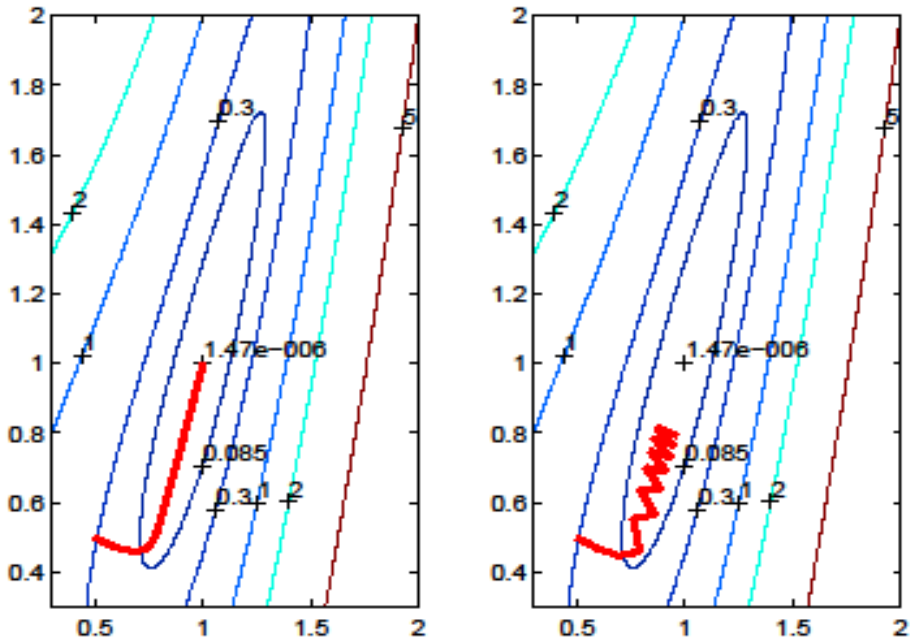


Рис. 4.5. Градиентный спуск с маленьким (левый рисунок) и большим шагом (правый рисунок)

Метод наискорейшего спуска

В этом методе s^k выбирается так, чтобы минимизировать функцию $f(x)$ в направлении спуска. Т.е. движение вдоль антиградиента происходит до тех пор, пока значение функции $f(x)$ убывает. Величина шага s^k определяется с помощью линейного поиска:

$$s^k = \arg\min f(x^k + sd^k)$$

В этом случае градиент в точке очередного приближения будет ортогонален направлению предыдущего спуска. Это означает, что движение от точки к точке будет происходить по взаимно-ортогональным направлениям (Рис. 4.6).

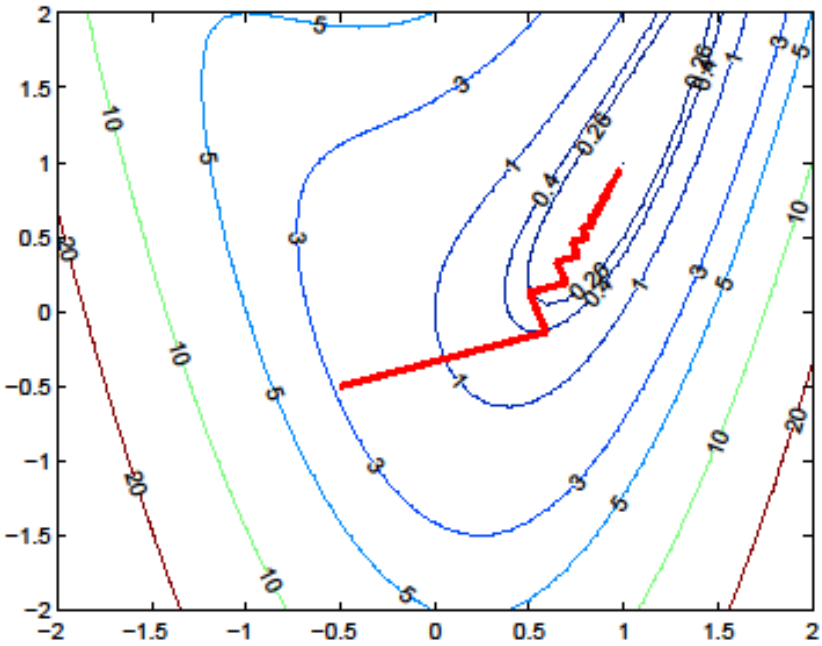


Рис. 4.6. Пример работы метода наискорейшего спуска

Методы второго порядка

Старейшим методом второго порядка, который используется для минимизации функции, является метод Ньютона. В этом методе выполняется квадратичная аппроксимация функции $f(x)$ в каждой точке x^k

$$f(x^{k+1}) \approx f(x^k) + (\nabla f(x^k)\Delta x^k) + 1/2\ (\Delta x^k) H(x^k)(\Delta x^k)^T,$$

где $H(x^k)$ - матрица Гессе, представляющая собой квадратную матрицу вторых частных производных функции f в точке x^k.

Для определения направления поиска, т.е. направления, в котором значение $f(x^{k+1})$ будет минимальным, выполняют дифференцирование f по каждой компоненте Δx и приравнивают к нулю полученные выражения. В результате получают соотношение:

$$\Delta x^k = -(H(x^k))^{-1}(\nabla f(x^k)^T,$$

где $(H(x^k))^{-1}$ - это матрица, обратная матрице Гессе в точке x^k.

Траектория поиска оптимума методом Ньютона показана на Рис. 4.7.

Существенным недостатком классической версии метода Ньютона является зависимость его сходимости от начального приближения для невыпуклых функций. Если начальная точка находится достаточно далеко от минимума, то метод может расходиться.

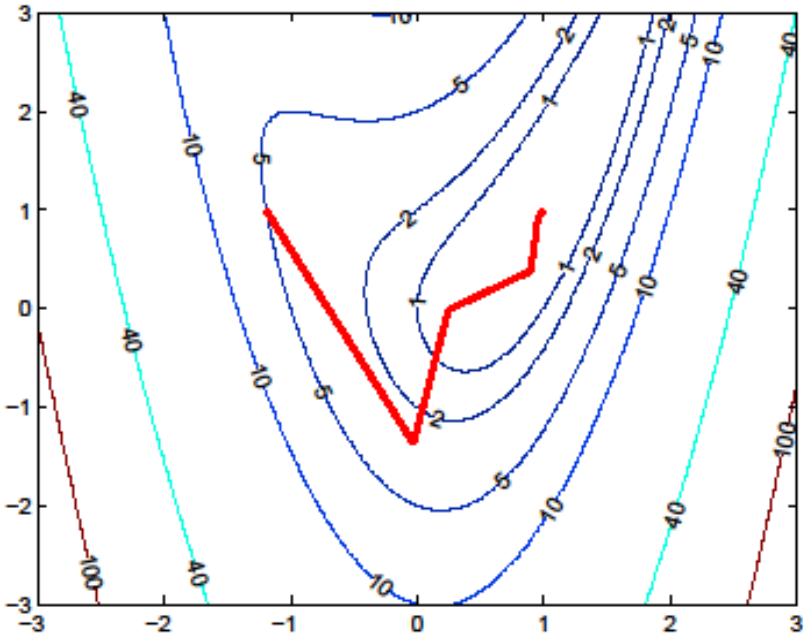


Рис. 4.7. Траектория поиска минимума методом Ньютона

Раздел 5. Нелинейная оптимизация с ограничениями

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*
2. *Писарук, Н. Н. Исследование операций / Н. Н. Писарук. — Минск : БГУ, 2013. — 294 с.*

Согласно информации из википедии, в теории оптимизации условия Каруша — Куна — Таккера (англ. Karush — Kuhn — Tucker conditions, KKT) — необходимые условия решения задачи нелинейного программирования. Чтобы решение было оптимальным, должны быть выполнены некоторые условия регулярности. Метод является обобщением метода множителей Лагранжа. В отличие от него, ограничения, накладываемые на переменные, представляют собой не уравнения, а неравенства.

Кун и Таккер обобщили метод множителей Лагранжа (для использования при построении критериев оптимальности для задач с ограничениями в виде равенств) на случай общей задачи нелинейного программирования с ограничениями, как в виде равенств, так и в виде неравенств.

Необходимые условия локального минимума для задач с ограничениями исследуются давно. Одним из основных остаётся предложенный Лагранжем перенос ограничений в целевую функцию. Условия КунаТаккера тоже выведены из этого принципа.

Тема 5.1. Необходимые условия оптимальности

Пусть рассматривается задача

$$\begin{aligned} & f(x) \to \min, \\ & g_i(x) \le 0, \quad i \in I = \{1, \dots, m\}, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{5.1}$$

где функции f и g_i ($i \in I$) непрерывно дифференцируемы. Обозначим через X множество решенийзадачи (5.1), т.е.

$$X = \{x \in \mathbb{R}^n : g_i(x) \leq 0,\ i \in I\}.$$

Предположим, что множество X непусто; однако допускаем, что оно может иметь пустую внутренность.

Точка $x^0 \in X$ есть локальный оптимум (минимум) задачи (5.1), если для некоторого числа $\epsilon > 0$ выполняется условие

$$f(x^0) \leq f(x) \quad \forall\, x \in X,\ \|x - x^0\| \leq \epsilon.$$

Если $x^0 \in X$ есть локальный оптимум задачи (5.1), то функция $f(x)$ не может убывать, когда x описывает дугу кривой (достаточно регулярной), выходящую из x^0 и содержащуюся в множестве решений X.

Такую дугу кривой будем называть допустимой и будем определять посредством непрерывно дифференцируемой функции $\phi : R_+ \to R^n$ параметра $\theta \geq 0$:

$$\phi(\theta) = [\phi_1(\theta), \ldots, \phi_n(\theta)]^T,$$

которая удовлетворяет условиям:

a) $\phi(0) = x^0$;

b) $\phi(\theta) \in X$ для достаточно малых $\theta > 0$.

Допустимым направлением в точке x^0 назовем вектор

$$y = \frac{d\varphi}{d\theta}(0) = \left[\frac{d\varphi_1}{d\theta}(0), \frac{d\varphi_2}{d\theta}(0), \ldots, \frac{d\varphi_n}{d\theta}(0)\right]^T,$$

касающийся некоторой дуги кривой $\phi(\theta)$, допустимой в x^0.

Необходимые условия Куна — Таккера

Теорема 5.1 (Куна — Таккера). Предположим, что все функции f и g_i ($i = 1,...,m$) непрерывно дифференцируемы и в точке $x^0 \in X$ выполняется условие выделения ограничений. Если x^0 есть точка локального минимума, то существуют такие числа $\lambda_i \geq 0$ ($i = 1,...,m$), что

$$\nabla f(x^0) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^0) = 0, \tag{5.2}$$

$$\lambda_i g_i(x^0) = 0, \quad i = 1, \ldots, m. \tag{5.3}$$

(Числа λ_i называются множителями Куна-Таккера.)

Точка $x^0 \in X$, которая удовлетворяет системе (5.2) и (5.3) называется стационарной точкой. Другими словами, теорема 5.1 утверждает, что при выполнении условия выделения ограничений локальные минимумы следует искать среди стационарных точек. Если все функции f и g_i ($i = 1,...,m$) выпуклы, то, все стационарные точки являются глобальными минимумами.

Тема 5.2. Достаточные условия оптимальности

Изучим достаточные условия оптимальности для задач следующего вида:

$$\begin{aligned} & f(x) \to \min, \\ & g_i(x) \leq 0, \quad i \in I = \{1, \ldots, m\}, \\ & x \in S \subseteq \mathbb{R}^n. \end{aligned} \tag{5.4}$$

В том случае, когда $S = R^n$, то вновь приходим к задаче (5.1). Но теперь множество S может быть даже дискретным ($S \subseteq Z^n$), и тогда получается задача целочисленного программирования.

Поставим в соответствие i-му ограничению ($i \in I$) неотрицательное действительное число, называемое множителем Лагранжа. Определим функцию Лагранжа по правилу:

$$L(x, \lambda) = f(x) + \sum_{i \in I} \lambda_i g_i(x).$$

Говорят, что точка $(\bar{x}, \bar{\lambda}) \in S \times \mathbb{R}^m_+$ есть седловая точка функции $L(x,\lambda)$, если

$$L(\bar{x}, \lambda) \leq L(\bar{x}, \bar{\lambda}) \leq L(x, \bar{\lambda}), \quad x \in S,\ \lambda \in \mathbb{R}^m_+. \tag{5.5}$$

Достаточное условие оптимальности.

Если пара $(\bar{x}, \bar{\lambda}) \in S \times \mathbb{R}^m_+$ есть седловая точка функции $L(x, \lambda)$, то $\bar{x}$, является глобальным минимумом в задаче (5.4).

Это условие - весьма общий результат, который применим к любым задачам вида (5.4): выпуклым и невыпуклым, с дифференцируемыми и недифференцируемыми функциями f и g_i, непрерывными и дискретными множествами S. Неудивительно, что имеются задачи, для которых функция Лагранжа не имеет седловых точек.

Задача выпуклого программирования — это задача (5.4), когда все функции f и g_i ($i \in I$) выпуклы, а множество S — также выпукло.

Теорема 5.1. Пусть все функции f и g_i ($i \in I$) выпуклы, множество S замкнуто и выпукло, и существует такая точка x ∈ S, что

$$gi(x) < 0, \quad i \in I. \tag{5.6}$$

Тогда если задача (5.4) имеет оптимальное решение $\bar{x}$, то существует такой вектор множителей $\lambda \in \mathbb{R}^m_+$, что $(x, \bar{\lambda})$, есть седловая точка функции Лагранжа $L(x,\lambda)$.

Теорема 5.2. Если в задаче (5.1) все функции f и gi f и g_i ($i \in I$) выпуклы и непрерывно дифференцируемы, то для того чтобы точка $\bar{x}$, $\in X$ была глобальным минимумом, необходимо и достаточно, чтобы в точке $\bar{x}$, выполнялись условия Куна — Таккера.

Раздел 6. Численные методы поиска условного экстремума

Материалы к лекциям данного раздела сформированы на основании следующих источников:

1. *Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.*

Методы штрафных и барьерных функций

Метод штрафных функций может применяться для решения задач условной оптимизации с ограничениями как в виде равенств, так и в виде неравенств. Основная идея метода заключается в том, чтобы аппроксимировать исходную задачу условной оптимизации некоторой последовательностью вспомогательных задач безусловной оптимизации, и затем решить задачи этой последовательности известными методами безусловной оптимизации. В методах штрафных функций ограничения задачи (все или некоторая их часть) отбрасываются, а к целевой функции добавляется штраф за их нарушение, т.е. слагаемое, которое принимает большие положительные значения в точках, не удовлетворяющих отброшенным ограничениям (для решения задач минимизации). Штрафная функция равна нулю в допустимой области.

Например, в случае ограничений в виде равенств, штрафная функция наиболее часто строится на основе квадратичного штрафа:

$$P(\mathbf{x}, r) = \frac{r}{2}\sum_{j\in\mathcal{E}} g_j^2(\mathbf{x}) + \frac{r}{2}\sum_{j\in\mathcal{I}} [-g(\mathbf{x}_j)]^2,$$

где

$$[a] = \max\{a, 0\}$$

P(x,*r)* - это штрафная функция, *r* - положительный параметр штрафа, *g*(x) – ограничения. Сходимость метода зависит от величины штрафа *r*. Чем больше *r,* тем выше скорость сходимости.

Метод барьерных функций (также известный как метод барьеров, или метод внутренних штрафов) добавляет слагаемое к исходной целевой функции, которое не позволяет генерируемым точкам выходить за пределы допустимой

области. Тем самым в методах барьерных функций исключен основной недостаток штрафных функций, связанный с тем, что при использовании штрафов аппроксимируемые точки не принадлежат множеству допустимых решений. Метод барьерных функций обычно используется в случае ограничений в виде неравенств.

В качестве барьерных функций можно использовать обратную и логарифмическую функции

$$B(\mathbf{x}, r) = r \sum_{j=1}^{m} \frac{1}{g_j(\mathbf{x})}, \quad B(\mathbf{x}, r) = r \sum_{j=1}^{m} \ln[-g_j(\mathbf{x})].$$

Обе эти функции определены и непрерывны внутри допустимого множества и стремяться к бесконечности при приближении к границе множества изнутри. Чем ближе значение параметра *r* к нулю, тем больше скорость сходимости метода. Однако с уменьшением *r* барьерная функция становится все более овражной. Поэтому принимать сразу r малым числом нецелесообразно.

1.2. Презентации

БНТУ

ФИТР

Numerical Analysis & Optimization

(Teaching Materials)

Pedro Lima, Vasily Volkov, Irina Kovaleva,
Irina Katkovskaya, Olga Zubko

Training. Tomsk. May. 2015

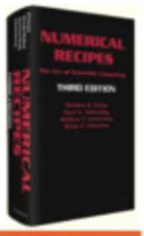
Our goal and intentions

- The proposed teaching materials are addressed to prospective engineers and scientists (first for users and then for developers).
- It is planed as a compact introduction to the subject-matter and practical guide in mathematical modelling and numerical simulations involving numerical analysis of differential equations and optimization problems.
- The main attention is paid to practical questions concerning the most useful numerical techniques including relatively new prospective approaches (FEM, multigrid methods and so on).

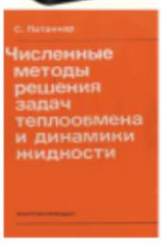
Examples of successful books in the area of Numerical Analysis

We intent to give practically useful recipes and explain how does it work .

Numerical recipes *3rd edition:*
The art of scientific computing
WH Press, WT Vetterling, SA Teukolsky, BP Flannery
- 2007 - books.google.com
Cited by **111 545** !!!



Numerical heat transfer and fluid flow.
S.Patankar.
CRC Press, 1980
In Russian 1984
Cited by **26 173** !!!



Content

ntroduction.

Numerical Methods for ODE

Initial Value Problems

uler Methods. Implicit and Explicit Schemes. Concept of Stability Runge-Kutta ethods. Adaptive Step Size Multistep Methods. Systems. Gear Methods BDF). Implicit Runge-Kutta Methods .

Boundary Value Problems . .

he Shooting Method . Finite-difference and Finite Element Methods for VP . Spectral Methods.

Numerical Methods for PDE

inite Difference Method . Time-dependent Heat Equation. Linear dvection Equation. Consistency, Stability and Convergence (Lax Theorem) . lliptic Equations in a Rectangular domain . Nonlinear Equations . Split-step ethods . Multigrid Methods .

ntroduction to the Finite Element Method Weak Form of the ifferential Problem. Finite Element Discretization . Galerkin Method . ixed Derivatives and Boundary Conditions. Discrete Finite Element odel. Assembly. Examples of applications of Finite Element Methods

Introduction

Scientific and technical computing.

Mathematical modeling. Numerical simulation. Numerical experiment as a modern technology of theoretic investigations. Mathematical model – numerical method computer program. Matlab: Language of technical computing.

Floating point arithmetic

Floating point numbers. Standard IEEE 745. Single and Double precision. Rounding error. Machine *epsilon* and *infinity*.

Vector and matrix Norms.

Vector norm axioms. Euclidean norm, p-norm, A-norm, ∞-norm. Equivalence of the norms. Induced matrix norm. Spectral matrix norm.

Stability and Conditioning of the problem.

Rounding and discretization error. Sensitivity of problems to perturbations of the input data. Stability. Error of linear perturbed systems. Residual and condition number

Sections and Subsections

- Statement of the problem
- Basic Concepts and ideas
- Theoretical Background
- Algorithmic details, implementation in standard software
- Advantages and disadvantages
- Use cases, preferences, restrictions
- Comparison of different approaches
- Examples
- Exercises

Sections in Ordinary Differential Equations

Initial Value problems for ODE

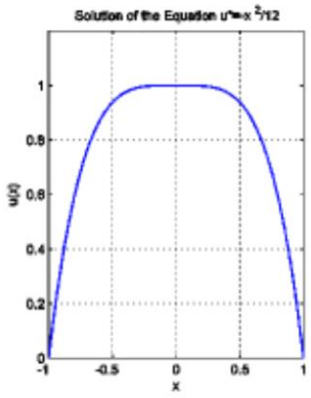
- Euler methods, Implicit and Explicit schemes.
- Runge-Kutta methods, adaptive step size.
- Multistep methods, Adams–Moulton and Adams–Bashforth methods.
- Stiff sets of Equations. Gear methods.

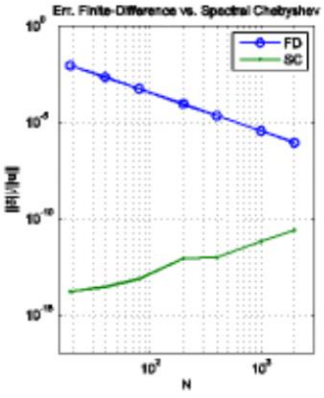
Two-point boundary value problem for ODE.

- The shooting method.
- Finite-difference and finite element methods.
- Spectral methods.

Accuracy Order, Smoothness and Convergence Rate

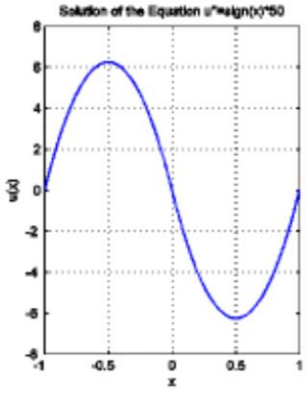
High order accuracy spectral methods demonstrate their advantages only in the case of sufficiently smooth solutions

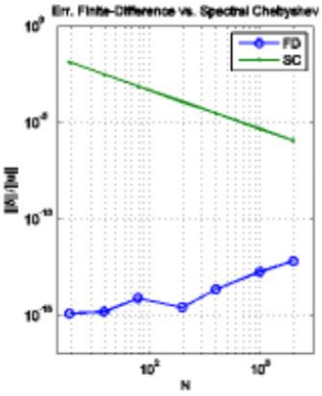




One more time about Smoothness and Convergence Rate

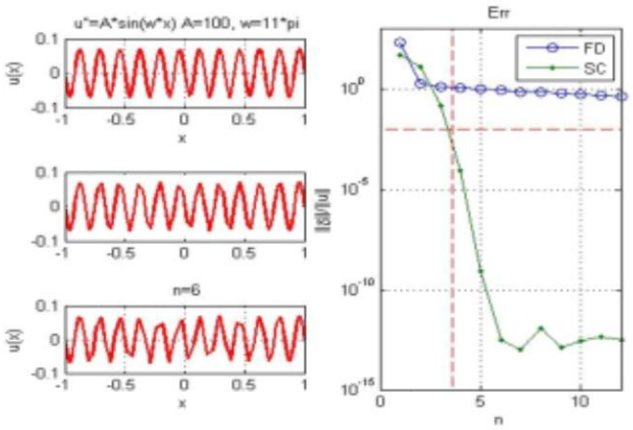
If the second derivative of the solution is discontinuous then advantages of the high order approximation spectral method are lost. Example:u''(x)=50 sign(x)





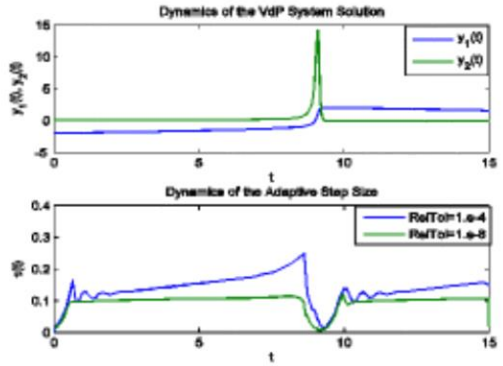
"Rules-of-Thumb" Step size for wave –like solutions

To get 1-5% accuracy we should use the step size h=λ/n, n=3.6 where λ is the wavelength of the wave-like solution



Adaptive Step Size in the embedded Runge-Kutta metod

How it is implemented in the Matlab function ode45. An Example with simulations of the Van der Pole oscillator

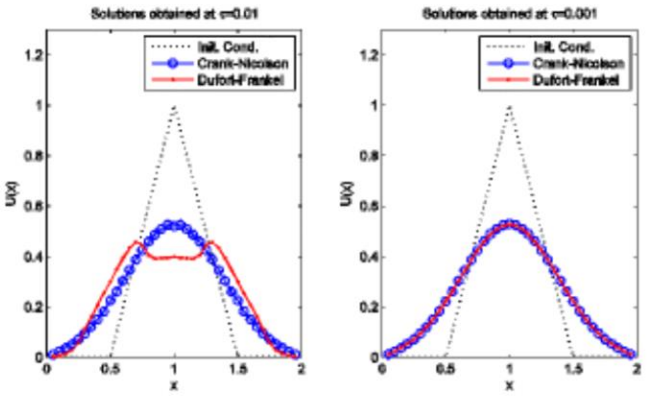


Section Finite difference method for PDE's

- Time-dependent Heat Equation.
- Linear Advection Equation.
- Consistency, Stability and Convergence
- (Lax Theorem) .
- Elliptic Equations in a Rectangular domain
- Nonlinear Equations . Split-step Methods
- Multigrid Methods .

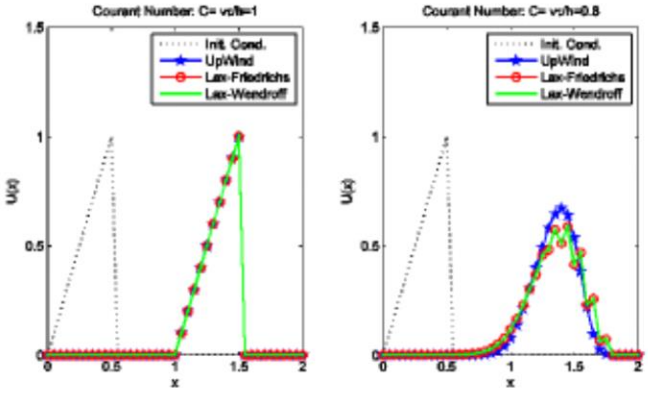
Time dependent Heat Equation

Crank-Nicolson vs. Dufort-Frankel schemes



Finite Difference Schemes for Advection Equation

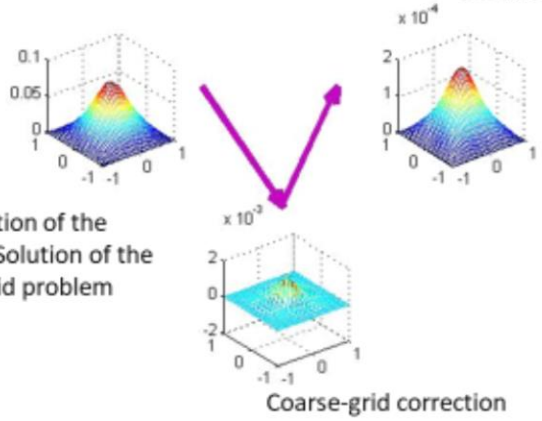
Upwind, Lax-Friedrichs and Lax-Wendroff schemes with different step size in time.



Multigrid method . 2D Poisson Equation

Pre-smoothing steps on the fine grid

Post-smoothing steps on the fine grid:



Computation of the residual. Solution of the coarse-grid problem

Coarse-grid correction

Section Introduction to the Finite Element Method

- Weak Form of the Differential Problem.
- Finite Element Discretization .
- Galerkin Method .
- Mixed Derivatives and Boundary Conditions.
- Discrete Finite Element Model. Assembly.
- Examples of applications of Finite Element Methods

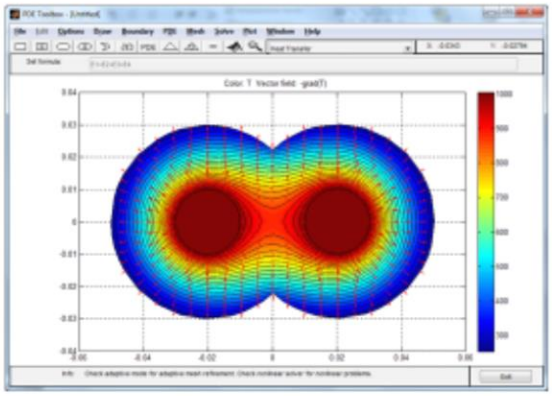
FEM in Matlab PDEtool box

Heating of the power cable at short-circuit

$$\nabla k(x,y)\nabla T = -\frac{I^2\sigma}{S^2}$$

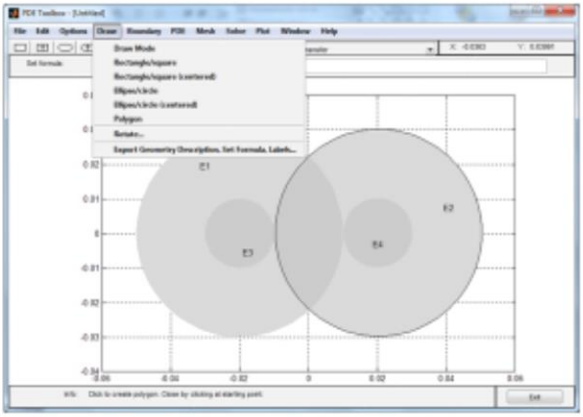
$$k(x,y)\frac{dT}{dn} = \alpha(x,y)(T - T_0)$$

I is the current,
S is the cross section area,
σ is the resistivity.



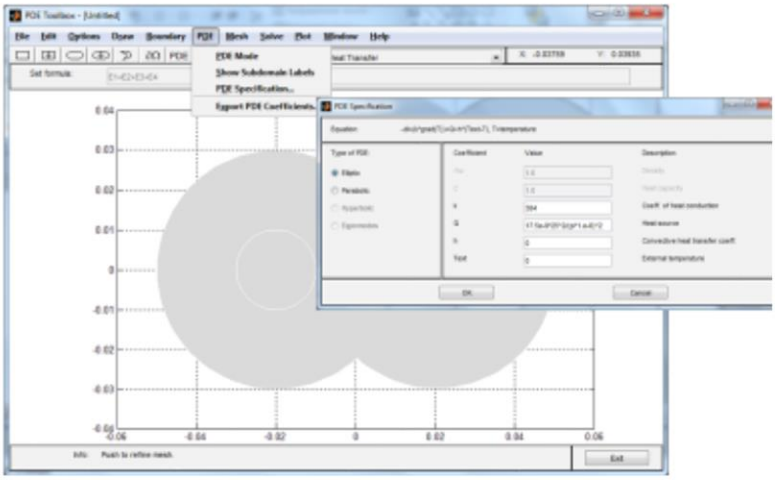
Definition of the geometry

The geometry of the problem can be easily specified using standard primitives (ellipse, rectangle, polygon)



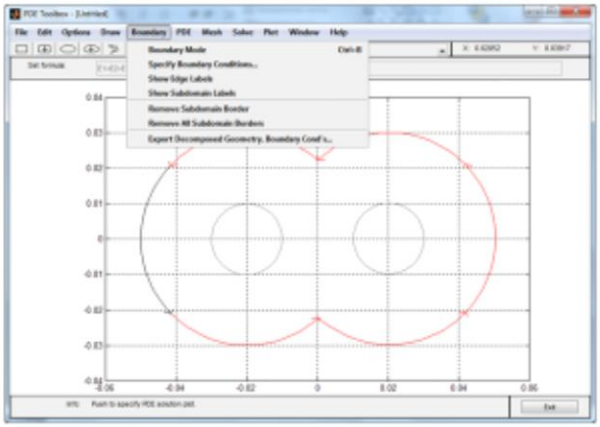
Specification of the coefficients

Coefficients of the problem can be specified for each subdomain separately in PDE Mode



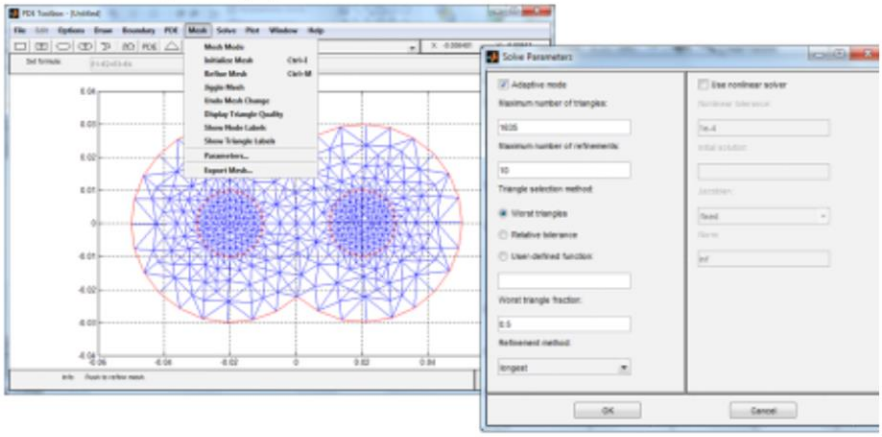
Specification of the boundary conditions

Specification of the boundary conditions is performed in the Boundary mode



Triangulations

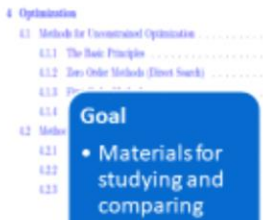
Triangular mesh is generated and refined in Mesh Mode. Furthermore, adaptive grid can be specified using corresponding option of the solver parameters.



Section Optimization

Preconditions

- Many books and textbooks on optimization
- Software for solving optimization problems



Goal

- Materials for studying and comparing different methods of optimization

Tasks

- A brief description of the basic methods
- Scripts of the basic methods for Matlab

Content

Zero Order Methods

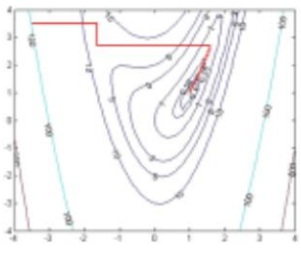
Zero Order Methods	Coordinate descent methods (Gauss method)
	Configuration or pattern method (Hooke and Jeeves method)
	Rosenbrock method
	Downhill method (Nelder-Mead method)
	Powell's method (Powell's conjugate direction method)
	Random search methods

Gauss method

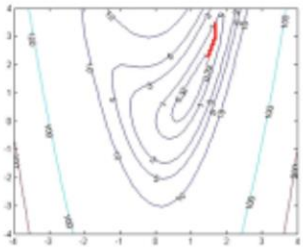
Scripts: function Gauss_bisection

function Gauss_goldensection

Simulation results for the Gauss method with bisection search

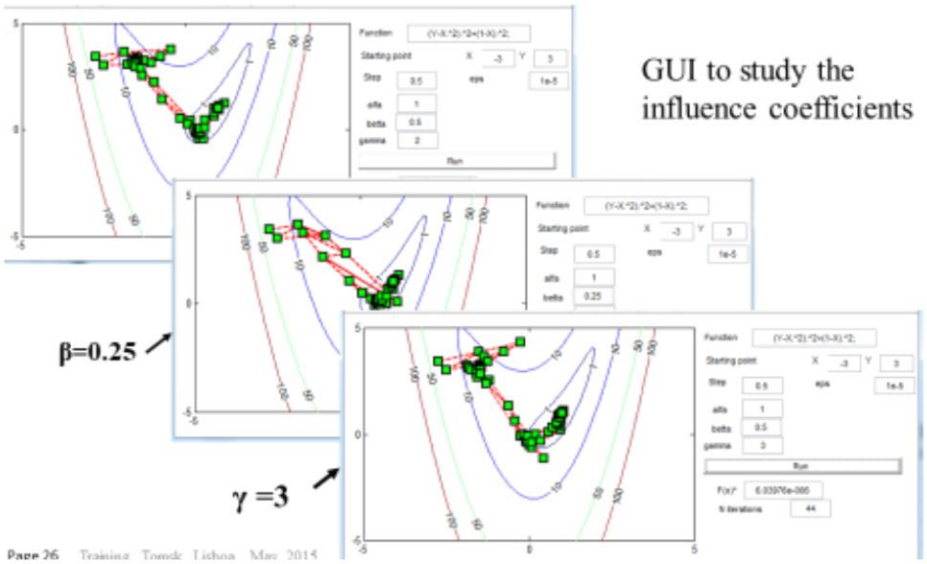


Successful



Unsuccessful

GUI for analysis of the Nelder-Mead method



GUI to study the influence coefficients

First Order Methods

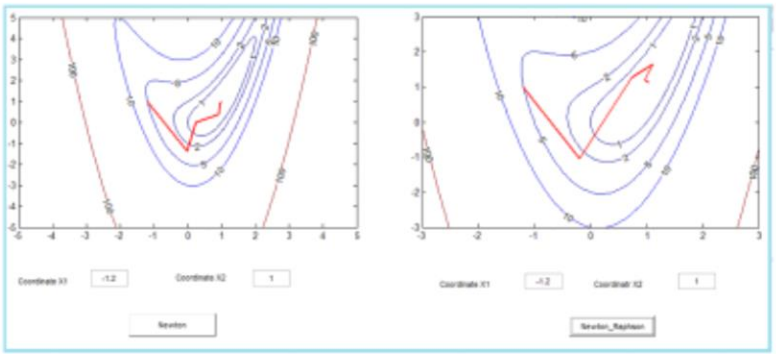
First Order Methods	Gradient descent with fixed step size
	Gradient descent with adaptively choose the step size
	Steepest descent method
	Gauss-Seidel method
	Fletcher-Reeves method
	Conjugate Gradient Methods

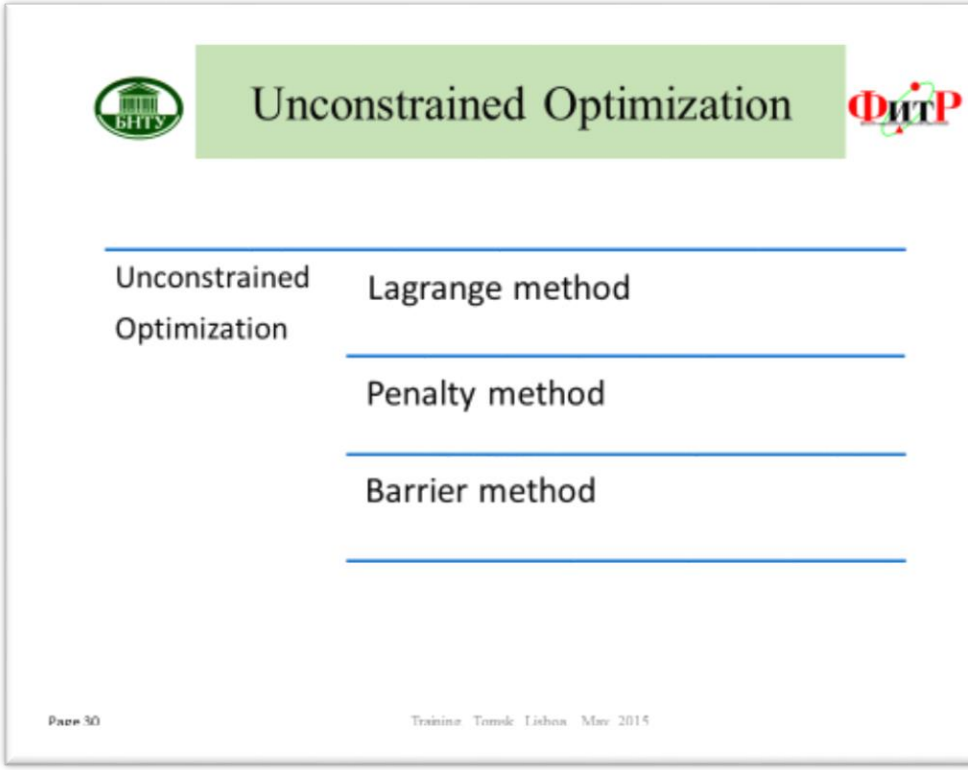
Second Order Methods

Second Order Methods	Newton's method
	Newton-Raphson method
	Marquardt method
	Quasi-Newton methods

Comparison of Newton's and Newton-Raphson methods

GUI developed to compare the effectiveness of Newton's and Newton-Raphson methods.




Unconstrained Optimization
Unconstrained Optimization
Lagrange method
Penalty method
Barrier method
Page 30
Training Tomsk Lisbon May 2015


БНТУ
ФИТР
Linear programming& Optimization
1-31 81 12 Прикладной компьютерный анализ данных 1

Linear programming

Theoretical aspects of linear programming	• Introduction • Statement of problem • Duality
Solution methods	• Graphic method • Simplex method
Applications of linear programming	• Transportation problem • Diet problem • Assignment problem • …

Diet problem (description)

Description: You are given a group of foods, their nutritional values and costs. You know how much nutrition a person needs. What combination of foods can you serve that meets the nutritional needs of a person but costs the least?

Food	Calories	Chocolate	Sugar (ounces)	Fat (ounces)	Cost (serving)
Fancy cake	400	3	2	2	$2.50 / cake
Chocolate ice cream	200	2	2	4	$1.00 / scoop
Coca-Cola	150	0	4	1	$1.50 / bottle
Pineapple cheesecake	500	0	4	5	$4.00 / slice

Diet problem (modeling)

x_1 = number of fancy cake to eat each day
x_2 = number of scoops of chocolate ice cream to eat each day
x_3 = number of bottles of Coke to drink each day
x_4 = number of pineapple cheesecake slices to eat each day
$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, and $x_4 \geq 0$

Objective is to minimize cost of food. Total daily cost is
Cost = (Cost of fancy cake) + (Cost of ice cream) + (Cost of Coca-Cola) +(Cost of cheesecake)= $2.5x_1 + x_2 + 1.5x_3 + 4x_4$

Constraint 1 - calorie intake at least 500
So constraint 1 is : $400x_1 + 200x_2 + 150x_3 + 500x_4 \geq 500$

Constraint 2 - chocolate intake at least 6 oz
So constraint 2 is : $3x_1 + 2x_2 \geq 6$

Constraint 3 - sugar intake at least 10 oz
So constraint 3 is : $2x_1 + 2x_2 + 4x_3 + 4x_4 \geq 10$

Constraint 4 - fat intake at least 8 oz
So constraint 3 is : $2x_1 + 4x_2 + x_3 + 5x_4 \geq 8$

Diet problem (LP-model)

we have to minimize

$$F(x) = 2.5x_1 + x_2 + 1.5x_3 + 4x_4$$

subject to the constraints

$$\begin{aligned} 400x_1 + 200x_2 + 150x_3 + 500x_4 &\geq 500 \\ 3x_1 + 2x_2 &\geq 6 \\ 2x_1 + 2x_2 + 4x_3 + 4x_4 &\geq 10 \\ 2x_1 + 4x_2 + x_3 + 5x_4 &\geq 8 \end{aligned} \quad \text{and} \quad \begin{aligned} x_1 &\geq 0 \\ x_2 &\geq 0 \\ x_3 &\geq 0 \\ x_4 &\geq 0 \end{aligned}$$

Diet problem (in matrix notation)

we want to

$$\text{minimize } f^T x \text{ subject to } Ax \geq b \text{ and } x \geq 0$$

where

$$A = \begin{bmatrix} 400 & 200 & 150 & 500 \\ 3 & 2 & 0 & 0 \\ 2 & 2 & 4 & 4 \\ 2 & 4 & 1 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 500 \\ 6 \\ 10 \\ 8 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \quad \text{and } f = \begin{bmatrix} 2.5 \\ 1 \\ 1.5 \\ 4 \end{bmatrix}$$

Simplex method (realization)

linprog

Solve linear programming problems

Equation

Finds the minimum of a problem specified by

$$\min_x f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

f, *x*, *b*, *beq*, *lb*, and *ub* are vectors, and *A* and *Aeq* are matrices.

Syntax

```
x = linprog(f,A,b)
x = linprog(f,A,b,Aeq,beq)
x = linprog(f,A,b,Aeq,beq,lb,ub)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options)
x = linprog(problem)
[x,fval] = linprog(...)
[x,fval,exitflag] = linprog(...)
[x,fval,exitflag,output] = linprog(...)
[x,fval,exitflag,output,lambda] = linprog(...)
```

Description

`linprog` solves linear programming problems.

```
>> A=-[400 400 150 500;3 2 0 0;...
      2 2 4 4;2 4 1 5];
>> b=-[500 6 10 8];
>> f= [2.5 1 1.5 4];
>> lb=[0 0 0 0];
>> x=linprog(f,A,b,[],[],lb)
Optimization terminated.

x =

    0.0000
    3.0000
    1.0000
    0.0000
```

Constrained and unconstrained optimization

Theoretical aspects of constrained and unconstrained optimization	• Introduction • The basic principles of an unconstrained optimization • The basic principles of a constrained optimization
Methods for unconstrained optimization	• Zeroth order methods (direct search methods) • First order methods • Second order methods
Methods for constrained optimization	• Penalization methods • Barrier methods • KKT conditions

Zeroth order methods

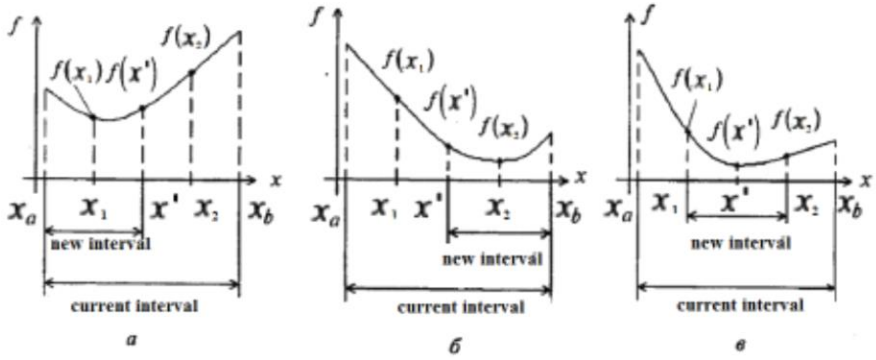
Part I: one-dimensional unconstrained optimization
- Bi-section search.
- Dichotomous search.
- Fibonacci method.
- Golden Section search (***fminbnd*** – **Matlab**).
- Quadratic interpolation.

Part II: multidimensional unconstrained optimization
- Method by Hooke and Jeeves.
- Method by Nelder and Mead (***fminsearch*** – **Matlab**).
- Rosenbrock's method .
- Powell's method.

Part III:
- Genetic algorithm (***ga*** – **Matlab**).
- Random search
- Generalized pattern search (GPS) algorithm, the generating set search (GSS) algorithm, and the mesh adaptive search (MADS) algorithm (***patternsearch***– **Matlab**).

Bi-section search (basic concepts)

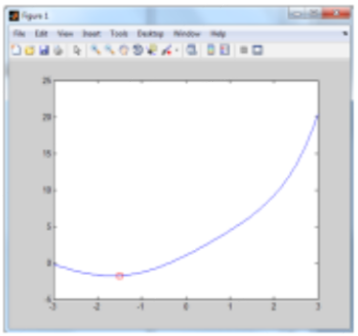
Idea: set initial interval $[x_a,x_b]$, on each iteration a half of an interval is excluded, the algorithm is based on analysis of function in three points dividing interval into four equal parts.



Bi-section search (realization)

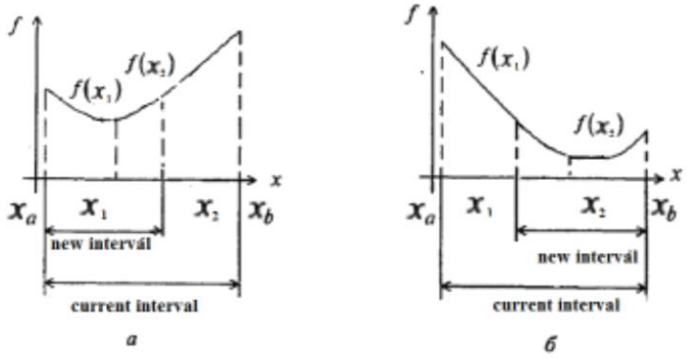
```
xa=str2num(input('a= ','s'));
xb=str2num(input('b= ','s'));
cur_a=xa;
cur_b=xb;
tochn=0.01;
aa = true;
while(aa)
  X_1=(cur_b+cur_a)/2;
  if(fun2(cur_a+(abs(X_1-cur_a)/2))<fun2(X_1))
     cur_b=X_1;
  else cur_a=X_1;
  end
  if (tochn>abs(cur_a-cur_b)) aa=false;
  end
end;
cur_a
x= xa:((abs(xb-xa))/100):xb;
plot(x,fun2(x),cur_a,fun2(cur_a),'--rs')
fun2(cur_a)
```

y=(2*sin(x))+exp(x)



Golden Section search (basic concepts)

Idea: set initial interval $[x_a, x_b]$, on each iteration a part of an interval is excluded, the algorithm is based on analysis of function in two points of the Golden Section.



Golden Section search (realization)

fminbnd

Find minimum of single-variable function on fixed interval

Equation

Finds a minimum for a problem specified by

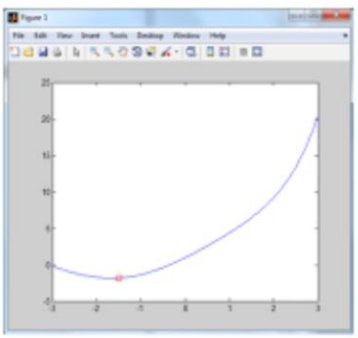
$$\min_x f(x) \text{ such that } x_1 < x < x_2.$$

x, x_1, and x_2 are scalars and f(x) is a function that returns a scalar.

Syntax

```
x = fminbnd(fun,x1,x2)
x = fminbnd(fun,x1,x2,options)
x = fminbnd(problem)
[x,fval] = fminbnd(...)
[x,fval,exitflag] = fminbnd(...)
[x,fval,exitflag,output] = fminbnd(...)
```

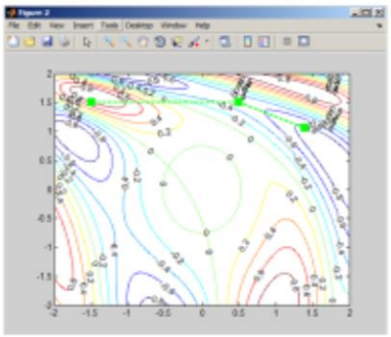
```
function f_1 = fun1(x)
f_1=2*sin(x) +exp(x);
end
x=fminbnd('fun1',-3,3)
```



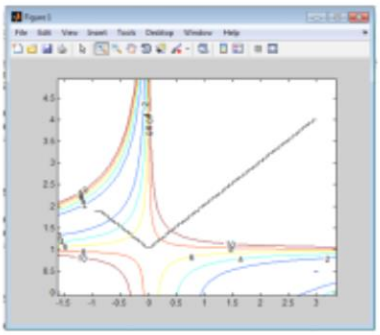
Rosenbrock's method and Method by Hooke and Jeeves

```
function zM = f2v1( v )
   zM = sin(1/2 * v(1)^2 + 1/4 * v(2)^2 + 3) *
cos(2*v(1) + 1 + exp(v(2)));
end
```

```
function f1= func1(x,y)
f1=(1.5-x.*(1-y)).^2+(2.25-x.*(1-y.^2)).^2;
end
```



Rosenbrock's method

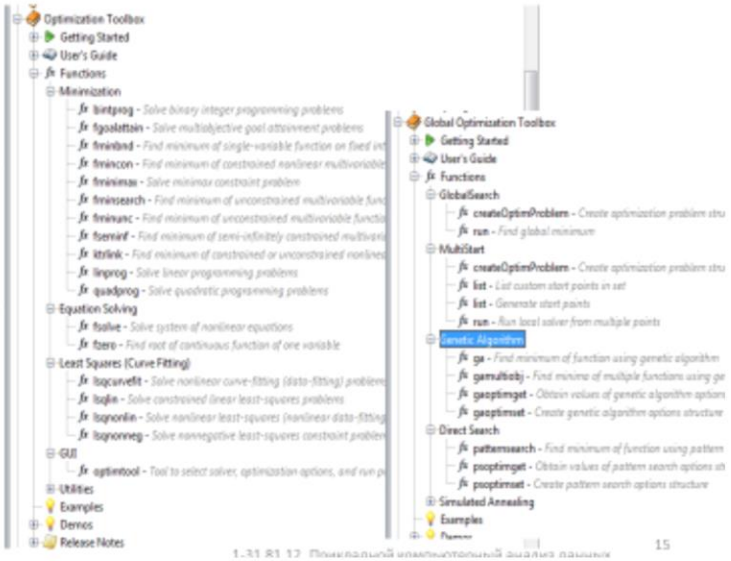


Method by Hooke and Jeeves

Development of additional Matlab libraries

New m-file

New m-function



1.3. Электронные образовательные ресурсы

1. https://www.matburo.ru/st_subject.php?p=dr

Ссылки на материалы по численным методам (вычислительной математике): учебники с теорией, практикумы с множеством разобранных примеров, видеоуроки.
Ссылки на математические программы, которые помогут в вычислениях, смотрите тут. Основные разделы с учебниками, лекциями и видео - тут.

2. https://rep.bntu.by/handle/data/27855

Численные методы для обыкновенных дифференциальных уравнений [Электронный ресурс]: учебно-методическое пособие для магистрантов специальности 1-31 81 12 «Прикладной компьютерный анализ данных»/ В.М.Волков, И.Л.Ковалева

3. https://matmetod-popova.narod.ru

Попова Н.В. Математические методы –- Электронный учебник, 2005 ВТК

4. https://matlab.exponenta.ru/optimiz/book_2/

А.Г.Трифонов. Постановка задачи оптимизации и численные методы ее решения.

5. www.pisaruk-9591.appspot.com/static/books/OR.pdf

Писарук, Н. Н. Исследование операций / Н. Н. Писарук. — Минск : БГУ, 2013. — 294 с.

6. http://matica.org.ua/metodichki-i-knigi-po-matematike/metody-optimizatcii-nekrasova-m-g

Некрасова М. Г. Методы оптимизации.

2. ПРАКТИЧЕСКИЙ РАЗДЕЛ

2.1 Задания для проведения лабораторных занятий

Упражнение 1.1

1.Сформулируйте задачу Коши для уравнений

$$\frac{du_k}{dt} = f_k(t, u_1, u_2, \ldots, u_N), \quad k = 1, \ldots, N,$$

$$\frac{dz}{dt} = v_z, \quad \frac{dv_z}{dt} = -\omega^2 z,$$

в эквивалентном виде для одного дифференциального уравнения второго порядка.

2. Сформулируйте следующую задачу Коши

$$-\frac{du^3}{dt^3} + \frac{d^2u}{dt^2} + \frac{du}{dt} = f(t, u), \quad t \in [t_0, T],$$

$$\frac{d^2u}{dt^2}\Big|_{t=t_0} = g_2, \quad \frac{du}{dt}\Big|_{t=t_0} = g_1, \quad u(t)\Big|_{t=t_0} = g_0,$$

в виде эквивалентной системы дифференциальных уравнений первого порядка.

Упражнение 1.2.

1. Исследовать устойчивость двухстадийного численного метода решения задачи Коши для уравнения

$$\frac{du}{dt} = \lambda u, \quad \lambda < 0.$$

состоящего в чередовании явной и неявной формул Эйлера

$$U(t_{k+1}) = U(t_k) + \tau f(t_k, U(t_k)), \quad t_k = k\tau, k = 0, 1, 2, \ldots$$

$$U(t_{k+1}) = U(t_k) + \tau f(t_{k+1}, U(t_{k+1})), \quad t_k = k\tau, k = 0, 1, 2, \ldots.$$

на нечетных и четных шагах соответственно.

2. Оцените локальную погрешность неявного метода Эйлера

$$U(t_{k+1}) = U(t_k) + \tau f(t_{k+1}, U(t_{k+1})), \quad t_k = k\tau, k = 0, 1, 2, \ldots.$$

и метода, рассмотренного в предыдущем упражнении.

3.Пусть задачи Коши была решена трижды с различными значениями шага численного интегрирования*:* $\tau = \tau_0$; $\tau = \tau_0/2$; и $\tau = \tau_0/4$, где τ_0 достаточно мало. Оцените погрешность метода и его порядок точности, используя упомянутые выше три приближенные решения, полагая асимптотическое поведение погрешности $|\delta| \simeq C\tau^p$ и считая точное решение неизвестным.

4. Воспроизведите численный эксперимент предыдущего пункта 3, используя численный алгоритм, рассмотренные в примере выше, используя $\tau_0 = 0.01$. Оцените фактическую погрешность численного метода, используя точное решение задачи и сравните результат с оценкой, полученной в предыдущем пункте 3.

Упражнение 1.3.

1. Внося необходимые коррективы в рассмотренном примере 1.2., замените программную реализацию методов Рунге – Кутты и Адамса – Башфорта на соответствующие стандартные функции MATLAB ode45 и ode15s, в которых реализованы указанные методы. Сравните эффективность стандартной

реализации рассмотренных методов, сопоставляя вычислительные затраты для достижения заданной точности

2. Используя тестовую задачу, рассмотренную в примерев 1.2 и упражнении 1.1., проверьте, как значения входного параметра RelTol в функциях ode45 и ode15s влияют на фактическую погрешностью численного решения.

Упражнение 1.4.

1. Сравните области устойчивости неявного и неявно-диагонального метода Рунге – Кутты,.

2. Сравните области устойчивости явных методов Рунге – Кутты четвертого и пятого порядков точности.

3. Вычислите коэффициенты неявного метода Гира 7-го порядка. Определите область устойчивости данного метода.

Сравните полученные результаты оценки эффективности на примерах систем Ван дер Поля и Лоренца.

Упражнение1. 5.

1. Сравните время решения задачи с использованием метода конечных разностей и конечных элементов в рассмотренном выше примере, используя функции Matlab tic и toc. Какой из методов представляется более эффективным?

2. Постройте конечно-разностный и конечно-элементный методы для численного решения задачи:

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{du}{dr}\right) = f(r), \quad x \in (0,1),$$

$$\left.\frac{du}{dr}\right|_{r=0} = 0, \quad u(1) = 0.$$

Упражнение 1.6.

1. Какое минимальное число узлов сетки требуется для достижения предельной (спектральной) точности при дифференцировании полинома 6-го порядка с использованием матрицы спектрального дифференцирования Чебышева. Проверьте ответ с помощью прямых вычислений, приготовив соответствующий пример произвольного полинома 6-го порядка.

Упражнение 2.1.

1. Используя программную реализацию метода Дюфорта – Франкела вычислите приближенное решени рассмотренной в примере задачи при различных значениях шага: h = 1.e − 2; 1.e − 3; 1.e − 4. Убедитесь, что решение схемы Дюфорта – Франкела не сходится при τ = h.

2. Внося необходимые изменения в программную реализацию метода Дюфорта – Франкела, замените схему Дюфорта – Франкела явной трехслойной схемой . Убедитесь, что данная схема не устойчива и не сходится при любых соотношениях шагов τ и h.

Упражнение 2.2.

1. Реализуйте программно схему Beam–Warming и сравните результаты расчетов на основе этой схемы и других методов при τ = h = 0.02, варьируя значение параметра v = 0.9; v = 1.1; v = 1.2.

2. Реализуйте программную схему КАБАРЕ. На основе численных экспериментов определите соотношение шагов τ и h при которых схема является точной.

Упражнение 2.3.

1.Исследовать устойчивость и согласованность схемы Лакса – Фридрихса. Сравните теоретические оценки и результаты численных экспериментов для скорости сходимости и условия устойчивости данной схемы

2. Оцените и сравните погрешности аппроксимации разностных схем) для уравнения переноса.

3. Исследуйте устойчивость и согласованность трехслойной разностной схемы для нестационарного уравнения теплопроводности.

Упражнение 3.1

Решите данные задачи симплекс методом

$$2x_1 + x_2 + x_3 + x_4 + 3x_5 \to \max,$$

а)
$$\begin{cases} x_1 - x_4 + x_5 = 2, \\ x_2 + x_4 + x_5 = 6, \\ x_3 + x_4 - 2x_5 = 0, \end{cases}$$

$$x_1 \geqslant 0, x_2 \geqslant 0, \ldots, x_5 \geqslant 0;$$

$$z = x_1 + x_2 \to \max,$$

б)
$$\begin{cases} x_1 + 3x_2 + x_3 = 15, \\ -2x_1 + 3x_2 + x_4 = 6, \\ x_1 - x_2 + x_5 = 0, \end{cases}$$

$$x_1 \geqslant 0, x_2 \geqslant 0, \ldots, x_5 \geqslant 0;$$

$$3x_1 - 7x_2 + 3x_4 - 9x_5 - 4x_6 \to \min,$$

в)
$$\begin{cases} x_1 + 5x_4 + 2x_5 - 3x_6 = 4, \\ x_2 + 3x_4 + x_5 - x_6 = 1, \\ x_3 - 8x_4 - 4x_5 + 5x_6 = 5, \end{cases}$$

$$x_1 \geqslant 0, x_2 \geqslant 0, \ldots, x_6 \geqslant 0;$$

Упражнение 3.2

Решить задачу параметрического линейного программирования. Объяснить полученные результаты.

1. Максимизировать

$L(x, \lambda) = (1 - \lambda)x_1 - (2 + 3\lambda)x_2 + x_3$

при условиях

$x_1 + 4x_2 + x_3 = 5$

$x_1 - 2x_2 + x_3 = -1$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

2. Максимизировать

$L(x, \lambda) = x_1 - (5 + \lambda)x_2 - (1 + \lambda)x_3 + x_4$

при условиях

$x_1 + 3x_2 + 3x_3 + x_4 = 3$

$2x_1 + 3x_3 - x_4 = 4$

$x_1, x_2, x_3, x_4 \geq 0$

$-\infty < \lambda < +\infty$

3. Максимизировать

$L(x, \lambda) = (1 - \lambda)x_1 + (1 + \lambda)x_2 - x_3$

при условиях

$x_1 - x_2 - x_3 = 4$

$x_1 + 15x_2 + x_3 = 2$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

4. Минимизировать

$L(x, \lambda) = -x_1 - (4 - \lambda)x_2 - (1 + 2\lambda)x_3$

при условиях

$4x_1 + 11x_2 + 3x_3 = 7$

$x_1 + x_2 - x_3 = 0$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

5. Максимизировать

$L(x, \lambda) = (1 - \lambda)x_1 + (3 - \lambda)x_2 + 5x_3$

при условиях

$x_1 + 2x_2 - x_3 = 2$

$x_1 - x_2 = 6$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

6. Минимизировать

$L(x, \lambda) = (-1 + \lambda)x_1 - (4 + \lambda)x_2 - (1 + 5\lambda)x_3$

при условиях

$x_1 - x_2 + x_3 = 3$

$2x_1 - 5x_2 - x_3 = 0$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

7. Максимизировать

$L(x, \lambda) = x_1 + (2 - 10\lambda)x_2 - (1 + \lambda)x_3$

при условиях

$x_1 + 7x_2 + 9x_3 = 8$

$x_1 + 3x_2 + 5x_3 = 4$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

8. Максимизировать

$L(x, \lambda) = (1 - \lambda)x_1 + (8 - 2\lambda)x_2 + (10 + \lambda)x_3$

при условиях

$x_1 - x_2 + 4x_3 = 2$

$x_1 - x_2 + 2x_3 = 0$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

9. Минимизировать

$L(x, \lambda) = -(1 + 2\lambda)x_1 - (1 - \lambda)x_2 - x_3$

при условиях

$-x_1 + x_2 + x_3 = 2$

$3x_1 - x_2 + x_3 = 0$

$x_1, x_2, x_3 \geq 0$

$-\infty < \lambda < +\infty$

10. Максимизировать

$L(x, \lambda) = (1 - \lambda)x_1 - 3x_2 - (5 + \lambda)x_3 - (1 + 2\lambda)x_4$

при условиях

$x_1 + 4x_2 + 4x_3 + x_4 = 5$

$x_1 + 7x_2 + 8x_3 + 2x_4 = 9$

$x_1, x_2, x_3, x_4 \geq 0$

$-\infty < \lambda < +\infty$

Упражнение 3.3

1. Фирма может выпускать два вида изделий, используя четыре вида оборудования. По нормативам для изготовления одного изделия первого вида оборудование первого, второго, третьего и четвертого вида придется занять соответственно на 2, 3, 4 и 1 день. Аналогично, для изготовления одного изделия второго вида те же станки придется занять в течение 6, 3, 0, 2 дней соответственно. Известен фонд времени оборудования первого вида равен 18 дням, второго вида — 15 дням, третьего и четвертого — соответственно 16 и 8 дням. Удельная прибыль от производства одного изделия первого вида составляет 600 руб., а от производства одного изделия второго вида — 900 руб. Составьте математическую модель поставленной задачи и решите эту задачу:

а) графическим методом; б) симплексным методом. Найдите оптимальный план производства, обеспечивающий фирме наибольшую прибыль, а также двойственные оценки каждого вида оборудования и дайте им экономическую интерпретацию.

2. Фирма выпускает из железа и проволоки трансформаторы двух видов. На один трансформатор первого вида тратится 5 кг железа и 3 кг проволоки, изготовление одного трансформатора второго вида требует затрат 3 кг железа и 2 кг проволоки. От реализации одного трансформатора фирма получает прибыль 600 и 500 руб. соответственно. Ежемесячно фирма закупает для 4,8 т железа и 3 т проволоки. Определите: а) сколько трансформаторов каждого вида необходимо выпускать для получения максимальной прибыли; б) каковы двойственные оценки ресурсов; в) по каким максимальным ценам имеет смысл закупать ресурсы; г) каковы остатки ресурсов после выполнения месячного оптимального плана производства; д) какие ресурсы образуют узкие места; е) какое оптимальное количество ресурсов, образующих узкие места, следует заказать дополнительно.

Упражнение 4.1.

1. Разработать GUI для сравнения эффективности работы различных методов линейного поиска в методе наискорейшего спуска.

2. Разработать GUI для сравнения эффективности работы различных методов линейного поиска на этапах исследующего поиска и поиска по образцу в методе Хука–Дживса.

3. Выполнить анализ влияния изменения значений коэффициентов α и β в методе Розенброка.

4. Сравнить эффективность работы различных методов линейного поиска в методе Пауэлла.

Упражнение 4.2.

1. Изучения влияния размера шага на сходимость метода градиентного спуска с постоянным шагом.

2. Разработать GUI для сравнения эффективности поиска минимума функций методами градиентного спуска с делением шага и линейного поиска с возвратом.

3. Выполнить сравнение эффективности различных методов расчета коэффициента β в методах сопряженных градиентов.

Упражнение 4.3.

1. Разработать GUI для сравнения эффективности работы методов Ньютона–Рафсона и Марквардта.

2. Сравненить эффективность работы методов Ньютона и Марквардта.

Упражнение 5.1

1. Выполнить исследование следующей оптимизационной задачи

$$(x_1 - 2/3)(x_2 - 1/2)(x_3 - 1/3) \to \min,$$
$$x_1^2 + x_2^2 + x_3^2 \le 1,$$
$$x_1 \ge 0,\ x_2 \ge 0, x_3 \ge 0.$$

а) Удовлетворяют ли точки $x^1 = (0,0,0)^T$ и $x^2 = (0,0,1)^T$ условиям Куна-Таккера;

б) Какие из точек x^1 и x^2 являются локальными минимумами рассматриваемой задачи?

2. Выполнить исследование следующей оптимизационной задачи

$$\begin{aligned} \mathrm{e}^{x_1+x_3} + x_2^4 + 4x_2 - (x_1 + x_3) &\to \min, \\ x_1^2 + x_2^2 + x_3^2 &= 1, \\ x_1^2 - x_2 &= 1. \end{aligned}$$

а) Найдите 3 допустимых решения (допустимых точки) с $x_3 = 0$;

б) Какие из этих 3-х точек удовлетворяют условиям Куна-Таккера, а какие являются локальными минимумами рассматриваемой задачи?

2.2. Программный блок

Пример 1.1 . «Устойчивость и сходимость»

```
%%% Устойчивость & Сходимость
%%% метод Эйлера для уравнения u'=-lambda*u
lambda = 1.5;
T = 10.
U0 = 2;
tau_0 = 2/lambda;
NN = round(T/tau_0)*10;
u = zeros(3,NN);
t = u;
n = 0;
t(:,1) = 0;

  u(:,1) = U0;
  for tau = [0.2, 0.8 1.1]*tau_0
  n = n+1;
  N = T/tau;
  U = U0;
  for m = 1:N-1
  U = (1-tau*lambda)*U;
  u(n,m+1) = U;
  t(n,m+1) = tau*m;
  end
  NN(n) = N;
  end
  plot(t(1,1:NN(1)),u(1,1:NN(1)),'.-',...
  t(2,1:NN(2)),u(2,1:NN(2)),'o-',...
  t(3,1:NN(3)),u(3,1:NN(3)),'o-')
  legend('\tau=0.2*\tau_0',...
  '\tau=0.8*\tau_0','\tau=1.1*\tau_0')
  xlabel('t')
  ylabel('U(t)')
  grid
```

Пример 1.2 . «Оценка эффективности методов решения ОДУ»

```
%Эффекктивность методов решения ОДУ
% Метод Рунге -- Кутты 4-го порядка (RC)
% Метод Адамса -- Башфорта 4-го порядка (AB)
% Метод предиктор -- корректор 4-го порядка (PC)
% Задача:   u''=-3u, t in [1,10];
% начальные условия: u(0)=0, u'(0)=3;
% точное решение: u(t)=sin(3t);
f = @(t,u)[u(2); -9*u(1)];
T = 10;
tau = 0.01;
u = [0;3];
N = T/tau;

ts = 0:tau:T;
U = zeros(2,N+1);
%Метод Рунге -- Кутты 4 порядка
tic
U(:,1) = u;
for m = 1:N
t = (m-1)*tau;
K1 = tau*f(t,u);
K2 = tau*f(t+tau/2,u+K1/2);
K3 = tau*f(t+tau/2,u+K2/2);
K4 = tau*f(t+tau,u+K3);
u = u+(K1+2*K2+2*K3+K4)/6;
U(:,m+1) = u;
end
RK_time = toc
B = tau*[-9/24;37/24;-59/24;55/24];
F = zeros(2,N+1);
u = [0;3];
U1 = zeros(2,N+1);
U1(:,1) = u;
F(:,1) = f(0,u);
```

```
% Метод Адамса -- Башфорта 4 порядка
tic
for m = 1:3
t = (m-1)*tau;
K1 = tau*f(t,u);
K2 = tau*f(t+tau/2,u+K1/2);
K3 = tau*f(t+tau/2,u+K2/2);
K4 = tau*f(t+tau,u+K3);
u=u+(K1+2*K2+2*K3+K4)/6;
U1(:,m+1) = u;
F(:,m+1) = f(t+tau,u);
end
for m = 4:N
u = u+F(:,m-3:m)*B;
t = m*tau;
F(:,m+1) = f(t,u);
U1(:,m+1) = u;
end
AB_time = toc

%Predictor-Corrector on the base of 4-th order
% Метод предиктор -- корректор 4-го порядка
A = tau*[1/24;-5/24;19/24;9/24;];
u2 = [0;3];
U2 = zeros(2,N+1);
U2(:,1) = u2;

F(:,1) = f(0,u2);
tic
for m=1:3
t = (m-1)*tau;
K1 = tau*f(t,u2);
K2 = tau*f(t+tau/2,u2+K1/2);
K3 = tau*f(t+tau/2,u2+K2/2);
K4 = tau*f(t+tau,u2+K3);
u2 = u2+(K1+2*K2+2*K3+K4)/6;
U2(:,m+1) = u2;
F(:,m+1) = f(t+tau,u2);
end
```

```
for m = 4:N
u22 = u2+F(:,m-3:m)*B;%Predictor
t = m*tau;
F(:,m+1) = f(t,u22);
u2 = u2+F(:,m-2:m+1)*A;%Corrector
F(:,m+1) = f(t,u2);
U2(:,m+1) = u2;
end
PC_time = toc
y = sin(3*ts); % Exact solution;
plot(ts,(U(1,:)-y),ts,U1(1,:)-y,ts,U2(1,:)-y,'LineWidth',1)
title(['Global Error of 4-th order ODEs solvers.
\tau=',num2str(tau,3)])
xlabel('t')
ylabel('\delta(t)=U(t)-u(t)')
legend('RK','AB','PC');
grid on
```

Пример 1.3. «Сравнение области устойчивости явного и неявного методов Рунге-Кутты»

```
%% область устойчивости явного и неявного методов РК
%% A,b,e коэффициенты явной четырехстадийной схемы РК %%%%%
%% A1,b1,e1 коэффициенты неявной двухстадийной схемы РК  %%%
x = -4:0.01:1; y = -3.5:0.01:3.5; [X,Y] = ndgrid(x,y);
A = [0 0 0 0; 1/2 0 0 0;0 1/2 0 0;0 0 1 0];
A1 = [1/4 (3-2*sqrt(3))/12; (3+2*sqrt(3))/12 1/4];
b = [1 2 2 1]/6; b1 = [1/2 1/2];
e = [1 1 1 1]; e1 = [1 1];
E = eye(4); E1 = eye(2);
R = zeros(size(X)); R1 = R;
for k = 1:length(x);
for m = 1:length(y);
z = X(k,m)+1i*Y(k,m);
R(k,m) = abs(1+z*b*(inv(E-z*A)*e'));
R1(k,m) = abs(1+100*z*b1*(inv(E1-100*z*A1)*e1'));
end
end
```

```
%% граница области устойчивости задана
%% уравнением: 1-|RE(z)|=0 и отображается изолинией
subplot(2,1,1), [C, H]= contourf(X,Y,1-R,[0 3])
colormap([0.9 0.9 0.99])
title('The Stability Domain of the 4-th order Explicit RK')
text(-0.7,0.9,'|R(z)|<0')
text(0.3,0.9,'|R(z)|>0')
xlabel('Re(z)')
ylabel('Im(z)')
grid on
subplot(2,1,2), [C, H]= contourf(100*X,100*Y,1-R1,[0 3]);
colormap([0.9 0.9 0.99]);
text(-90,90,'|R(z)|<0');
text(30,90,'|R(z)|>0')
xlabel('Re(z)')
ylabel('Im(z)')
title('The Stability Domain of the 4-th order Implicit RK')
grid on
```

Пример 1.4. «Зависимость погрешности спектрального метода дифференцирования Фурье от гладкости дифференцируемой функции»

```
% Погрешность спектрального метода дифференцирования Фурье
%  в зависимости от гладкости дифференцируемой функции
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = 128/1;
h = 2*pi/N;
x = -pi:h:pi-h; %Note, the last point is omitted
d = 1i*fftshift(-N/2:N/2-1);
%%%% гладкая функция u_1(x)%%%%%%%%%%%%%%%%%%%%%%%%%
u1 = exp(-8*x.^4);
%%%% не гладкая функция u_1(x)%%%%%%%%%%%%%%%%%%%%%
u2(1:N/2) = exp(-8*x(1:N/2).^4);
u2(N/2+1:N) = exp(-8*x(N/2+1:N).^2);
%%% точные производные %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
du1 = -32*x.^3.*exp(-8*x.^4);
du2(1:N/2) = -32*x(1:N/2).^3.*exp(-8*x(1:N/2).^4);
du2(N/2+1:N) = -16*x(N/2+1:N).*exp(-8*x(N/2+1:N).^2);
```

```
%%% Спектральные производные %%%%%%%%%%%%%%%%%%%%%%%
Du1 = ifft(d.*fft(u1));
Du2 = ifft(d.*fft(u2));
%%%%error of the spectral differentiation %%%%%%
err1 = abs(Du1-du1);
err2 = abs(Du2-du2);
m=1:4:length(x);
semilogy(x(m),err1(m),'o-',x(m),err2(m),'.-','LineWidth',1)
legend('smooth','non smooth')
xlabel('x')
ylabel('Погрешность спектрального дифференцирования')
grid
```

Пример 2.1. «Уравнение Пуассона с краевыми условиями Дирихле»

```
%%    Уравнение Пуассона  %%%%%%%%%%%%%%%%%%
%%       с краевыми условиями Дирихле  %%%%%%
L = 2;
n  =29;
%%%%   Построение сетки  %%%%%%%%%%%%%%%%%%%
%% граничные точки не входят в сетку %%%%%%%
h  = L /(n+1);
x =  -L/2+h :h :L /2-h ;
[Y,X] = ndgrid(x,x);
F =  10*sin(2*pi*X/L).*sin(2*pi*Y/L);
A=-gallery('poisson',n )/h^2 % Poisson matrix
%%  Собственные значения матрицы Пуассона  %%
g = - 4/h^2*(sin(pi*(1:n)*h/(2*L))).^2
[g_x,g_y] = ndgrid(g,g);
Lambda=g_x+g_y;
%%  Нулевые краевые условия  %%%%%%%%%%%%%%%
%% Метод конечных разностей %%%%%%%%%%%%%%%%
U0 = A\F(:);
U0 = reshape(U0,n,n);
```

```
%%      Фурье метод реализации РС    %%%%%%%%

F0 = reshape(F,n,n);
%% Дискретное sin-Фурье преобразования  dst %
%  применяется дважды (к столбцам и строкам)%
f = dst(F0);            % по столбцам
f = dst(f.').';         % по строкам
UU=f./Lambda
%% обратное  sin-Фурье преобразования  idst %
UU = idst(UU);          % по столбцам
UU = idst(UU.').';      % по строкам
%%  ненулевые граничные условия    %%%%%%%%%%
b1 = -L/2;
F(1,:) =F(1,:) - b1/h^2;          % при x=-L/2
b2 = x;
F(:,1) = F(:,1) - b2(:)/h^2;      % при y=-L/2
b3 =  L/2;
F(end,:) = F(end,:) - b3/h^2;     % при x= L/2
b4 = x;
F(:,end) =F(:,end) - b4(:)/h^2;   % при y= L/2

%%    Метод конечных разностей %%%%%%%%%%%%%%
U = A\F(:);
U = reshape(U,n,n);

%%      Фурье метод реализации РС    %%%%%%%%
F0 = reshape(F,n,n);
f = dst(F0);
f = dst(f.').';
Uf=f./Lambda
Uf = idst(Uf);
Uf = idst(Uf.').';
figure('Position',[403 246 760 300])
subplot(1,2,1), contour(X,Y,U0,17);
subplot(1,2,1), mesh(X,Y,U0);
xlabel('x'); ylabel('y');
title('Zero boundary conditions')
subplot(1,2,2), contour(X,Y,U,17);
subplot(1,2,2), mesh(X,Y,U);
title('Non Zero boundary conditions')
xlabel('x'); ylabel('y');
```

Пример 2.2. «Многосеточный метод для уравнения Пуассона»

```
%% Многосеточный метод для уравнения Пуассона %%%
%%  с краевыми условиями Дирихле  %%%%
clear
L = 2;
n = 21;  n2 = 11;
n=27;  n2 = 14;
% n=51;  n2 = 26;
%%%% генерация сеток  %%%%%%%%%%%%%%%%%%%%
h = L/(n-1);   % шаг мелкой сетки
H = L/(n2-1);  % шаг грубой сетки
x  = -L/2 :h :L/2;   [X,Y] = meshgrid(x,x);   %   мелкая сетка
x2 = -L/2 :H :L/2; [X2,Y2] = meshgrid(x2,x2); %   грубая сетка
mask = ones(size(X));
mask(1:2:end,1:2:end) = mask(1:2:end,1:2:end)*0;
down=find(mask(2:n-1,2:n-1) == 0);
%%%  характеризация мелкой сетки       %%%%%%%%%%%%%%%%%%%%%%%
Ah = -gallery('poisson',n-2 )/h^2;      % матрица Пуассона
L_Ah = 4/h^2*(sin(pi*h*(1:n)/4)).^2+...
4/h^2*(sin(pi*h*(1:n)/4)).^2;  % собственные значения матрицы
tau_0 = 2/(L_Ah(1)+L_Ah(end));  % опт. итерационный параметер
%%%  характеризация грубой сетки       %%%%%%%%%%%%%%%%%%%%%%%
AH = -gallery('poisson',n2-2 )/H^2;  % матрица Пуассона
L_AH = 4/H^2*(sin(pi*(1:n2-1)/n2/2)).^2+...
4/H^2*(sin(pi*(1:n2-1)/n2/2)).^2;%собственные значения матрицы
Tau_0 = 2/(L_AH(1)+L_AH(end))
tau = tau_0*3/4;  Tau = Tau_0*3/4;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Fh = (cos(pi*X/2).*cos(pi*Y/2)).^1;
Err0 = zeros(size(Fh));
Err1 = Err0;
Err  = Err0;
U = zeros(size(Fh)); U0 = Fh(2:end-1,2:end-1);
FH = reshape(Fh(down),n2-2,n2-2); Fh = Fh(2:end-1,2:end-1);
%%                точное решение     %%%%%%%%%%%%%%%%%%%%%%
UU=Ah\Fh(:);  U0 = reshape(UU,n-2,n-2);
U(2:end-1,2:end-1)=U0;
UT=AH\FH(:);
% subplot(2,2,1), mesh(X,Y,reshape(U,n,n));
% title('Exact Solution')
%%     Метод простой итерации     %%%%%%%%%%%%%
KK=800;
Uh = zeros(size(Fh(:)));UH = zeros(size(FH(:)));
Err_SIM=zeros(KK,1);
for k=1:KK
Uh=Uh-tau_0*(Fh(:)-Ah*Uh);
Err_SIM(k)=norm(Uh-U0(:))/norm(U(:));
UH=UH-Tau_0*(FH(:)-AH*UH);
Err_SIM_H(k)=norm(UH-UT(:))/norm(UT(:));
end
%%    операции сглаживания    %%%%%%%%%%%%%%%%%%
Uh = zeros(size(Fh(:)));  % начальное приближение
K = 5; % 3-5 итераций сглаживания в нвчале и конце
for k=1:K
Uh = Uh - tau*(Fh(:) - Ah*Uh);
end
Err0(2:end-1,2:end-1) = reshape(Uh(:),n-2,n-2) -U0;
subplot('position',[0.05 0.53 0.42 0.40]), mesh(X,Y,(Err0));
title('Pre-smoothing Error')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% расчет невязки на грубой сетке %%%%%%%%%%%%
rh = -Fh(:)+Ah*Uh; %невязка на мелкой сетке
rH = rh(down);     %проекция на грубую сетку
eH = AH\rH;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%  Коррекция с грубой сетки   %%%%%%%%%%%%%%%%
eH = reshape(eH,n2-2,n2-2);
EE = zeros(size(X2));
EE(2:end-1,2:end-1) = eH;   % продолжение на мелкую сетку
EE = interp2(X2,Y2,EE,X,Y); % интерполяция
eh = EE(2:end-1,2:end-1);
eh = eh(:);
Uh = Uh-eh(:);
```

```
Err1(2:end-1,2:end-1)=reshape(Uh,n-2,n-2)-U0;
subplot('position',[0.30 0.03 0.42 0.40 ]), mesh(X,Y,Err1);
limits=axis;
title('Coarse-Grid Correction Error')
%% итерации сглаживания  %%%%%%%%%%%%%%%%%%%%%%%%%%%%
tau = 3/4*2/(L_Ah(1)+L_Ah(end))
for k = 1:K
Uh = Uh-tau*(Fh(:)-Ah*Uh);
end
Err(2:end-1,2:end-1) = reshape(Uh,n-2,n-2)-U0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot('position',[0.53 0.53 0.42 0.40 ]), mesh(X,Y,Err);
title('Post-Smoothing Error')
axis(limits);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure
out = 1:20:KK;
semilogy(out,Err_SIM(out),'.-',out,Err_SIM_H(out),'o-')
xlabel('Iterations');
ylabel('Relative Accuracy')
legend('\omega_h','\omega_{2h}')
title('Convergence of SIM'); grid
```

Пример 4.1 «Метод Гаусса с методом деления пополам для линейного поиска»

```
function Gauss_bisection = Gauss_bisection()
t = input ('Enter the left grid boundary:','s'); Ax = str2num(t);
t = input ('Enter the right grid boundary:','s'); Bx = str2num(t);
t = input ('Enter the upper grid boundary:','s'); A1y = str2num(t);
t = input ('Enter the bottom grid boundary:','s');B1y = str2num(t);
t = input ('Enter x_0:','s');x_0 = str2num(t);
t1 = input ('Enter y_0:','s');y_0= str2num(t1);
t1 = input ('Enter eps:','s');eps= str2double(t1);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[X,Y] = meshgrid(Ax:.02:Bx);
Z=(Y-X.^2).^2 + (1 - X).^2;
[C,h]= contour(X,Y,Z,[0,0.1,1,3,5,10,100]);
clabel(C,h)
v = [x_0, y_0];
min_z = fun2(v);
zl = min_z;
min_x = x_0;
min_y = y_0;
min_con=0;i=1;
count_steps = 0;
while abs(zl-min_con)>eps*eps*eps
count_steps = count_steps + 1;
A = Ax;
B = Bx;
A1 = A1y;
B1 = B1y;
y_0 = min_y;
x_0 = min_x;
zl = min_z;
while (B - A) > 3*eps
m = A + (B - A)/4;
n = B - (B - A)/4;
x_sr = (B + A)/2;
v1 = [m, y_0];
v2 = [x_sr, y_0];
v3 = [n, y_0];
if fun2(v1) < fun2(v2)
B = x_sr;
end
if fun2(v1) > fun2(v2)
if fun2(v3) < fun2(v2)
A = x_sr;
end
if fun2(v3) >= fun2(v2)
A = m;
B = n;
end
end
end
while A < B
v4 = [A, y_0];
Z = fun2(v4);
if min_z > Z
min_z=Z;
```

```
min_x=A;
end
A = A + eps*eps;
end
Xl = [x_0,min_x];
Yl = [y_0, y_0];
Zl = [zl, min_z];
line([x_0,min_x],[y_0, y_0],[zl, min_z],'Color','g','LineWidth',3)
zl = min_z;
while (B1 - A1) > 3*eps
m = A1 + (B1 - A1)/4;
n = B1 - (B1 - A1)/4;
y_sr = (B1 + A1)/2;
v1y = [min_x,m];
v2y = [min_x, y_sr];
v3y = [min_x, n];
if fun2(v1y) < fun2(v2y)
B1 = y_sr;
end
if fun2(v1y) > fun2(v2y)
if fun2(v3y) < fun2(v2y)
A1 = y_sr;
end
if fun2(v3y) >= fun2(v2y)
A1 = m;
B1 = N;
end
end
end
while A1 < B1
v4y = [min_x, A1];
Z = fun2(v4y);
if min_z > Z
min_z=Z;
min_y=A1;
end
A1 = A1 + eps*eps;
end
min_con=min_z;
Xl1 = [min_x,min_x];
Yl1 = [y_0, min_y];
Zl1 = [zl, min_z];
line(Xl1,Yl1,Zl1,'Color','g','LineWidth',3)
i = i+1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Пример 4.2. «Метод Розенброка»

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function result = rosenbrok(x0,num,a,b,threshold,dx0, myFunc)
%x0 - starting point; num - the maximum number of failed steps per
iteration;
%xmax, xmin - the left and the right grid boundaries;
%threshold - stopping criterion
%dx0 - the length of the initial steps; myFunc -function reference
failCount = 0;    isEndWhile = 0;    directions = [1 0; 0 1];
lambda = [0 0]; dx = dx0; xCur = x0; yCur = myFunc(x0(1), x0(2));
xk = x0; yk = yCur; vy = [ xk yk ];
while isEndWhile == 0
yPrev = myFunc(xCur(1), xCur(2));
dxPrev = dx;
for i = 1:2
xNext = xCur;
for j = 1:2
xNext(j) = xNext(j) + directions(i,j) * dx(i);
end
yNext = myFunc(xNext(1), xNext(2));
if(yNext < yCur
xCur = xNext;
yCur = yNext;
lambda(i) = lambda(i) + dx(i);
dx(i) = dx(i) * a;
vy = [vy; xCur yCur];
else
```

```
dx(i) = dx(i) * b;
end
end
if(yCur == yPrev)
failCount = failCount + 1;
if(yCur < yk)
dist = 0;
for i = 1:2
dist = dist + (xk(i) - xCur(i))^2;
end
if(sqrt(dist) < threshold)
isEndWhile = 1;
else
xk = xCur;
yk = yCur;
end
directions = GramN(directions, lambda);
lambda = [0 0];
dx = dx0;
failCount = 0;
else
if(failCount >= num)
dist = 0;
```

```
for i = 1:2
dist = dist + (xk(i) - xCur(i))^2;
end
if(sqrt(dist) < threshold)
isEndWhile = 1;
else
xk = xCur;
yk = yCur;
end
directions = GramN(directions, lambda);
lambda = [0 0];
dx = dx0;
failCount = 0;
else
if(abs(dxPrev) < threshold)
isEndWhile = 1;
end
end
end
end
end
[x ,y] = meshgrid(-3:0.01:3, -3:0.01:3);
z = myFunc(x,y);
[C, h] = contour(x, y, z,[0, 1, 10,100,1000]);
clabel(C, h);
shading interp
hold on
plot3(vy(:,1),vy(:,2),vy(:,3),'r','LineWidth',2,'MarkerSize', 5);
hold off
disp(vy);
result = [ xCur yCur ];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

3. РАЗДЕЛ КОНТРОЛЯ ЗНАНИЙ

3.1. Контрольные вопросы к разделу 1.

1. Устойчивые и неустойчивые дифференциальные уравнения и численные методы.
2. Одношаговые численные методы решения задачи Коши.
3. Одношаговые численные методы решения задачи Коши. Метод Эйлера.
4. Одношаговые численные методы решения задачи Коши. Методы Рунге–Кутты.
5. Оценка погрешности метода Рунге-Кутты.
6. Многошаговые численные методы решения задачи Коши.
7. Многошаговые численные методы решения задачи Коши. Метод Адамса.
8. Многошаговые численные методы решения задачи Коши. Метод Адамса–Башфорта.
9. Многошаговые численные методы решения задачи Коши. Метод Адамса–Моултона.
10. Жесткие системы.
11. Метод Гира.
12. Устойчивость численных методов решения задачи Коши. Неявные методы.
13. Формулировка краевой задачи.
14. Метод конечных разностей для решения краевых задач.
15. Понятие разностной схемы.
16. Метод конечных элементов для решения краевых задач.
17. Спектральные методы коллокации.
18. Метод Фурье.
19. Спектральный метод Чебышева.
20. Оценка гладкости решения.

3.2. Контрольные вопросы к разделу 2.

1. Нестационарное одномерное уравнение теплопроводности
2. Краевые условия Дирихле.
3. Типовые сеточные шаблоны для уравнения теплопроводности.
4. Разностная схема с весами, схема Кранка-Николсона, схема Дюфорта-Франкела.
5. Линейное уравнение переноса.
6. Типовые шаблоны разностных схем для уравнения переноса.
7. Разностные схемы для уравнения переноса.
8. Исследование согласованности и устойчивости разностных схем.
9. Сходимость, скорость сходимости. Теорема Лакса.
10. Уравнения в частных производных эллиптического типа.
11. Типовые шаблоны для аппроксимации двумерных эллиптических уравнений в частных производных.
12. Нелинейные уравнения.
13. Нелинейное уравнение переноса.
14. Консервативность нелинейных разностных схем.
15. Монотонность разностного метода.
16. Метод переменных направлений.
17. Методы расщепления, или методы дробных шагов.
18. Метод простой итерации.
19. Метод последовательной верхней релаксации.
20. Многосеточный метод.
21. Слабая формулировка дифференциальной задачи
22. Конечно-элементная дискретизация
23. Метод Галеркина
24. Смешанные производные и граничные условия
25. Дискретная конечно-элементная модель. Сборка

26. Примеры программного обеспечения для метода конечных элементов.

3.3. Контрольные вопросы к разделу 3.

1. Сформулировать математические модели «классических» задач линейного программирования.
2. Формулировка задачи линейного программирования в каноническом виде.
3. Правила приведения задачи линейного программирования к каноническому виду.
4. Графическое решение задач линейного программирования.
5. Симплекс-метод. Ограничения типа равенств.
6. Симплекс-метод. Ограничения типа неравенств.
7. Двойственность
8. Параметрическое линейное программирование.
9. Примеры задач целочисленного линейного программирования.
10. Методы решения задач целочисленного программирования.
11. Постановка транспортной задачи и стратегии решения.
12. Методы нахождения начального плана перевозок.
13. Метод потенциалов

3.4. Контрольные вопросы к разделу 4.

1. Что такое математическая модель объекта оптимизации?
2. Сформулируйте математическую постановку задачи оптимизации.
3. Дайте определение оптимального решения задачи оптимизации.
4. Сформулируйте идею методов прямого поиска нулевого порядка.
5. Каким образом выбирают направления и параметр шага в методе Гаусса Зейделя?
6. В чем заключается этап исследующего поиска в методе Хука-Дживса?

7. Как выбирается ускоряющий множитель в этапе поиска по образцу в методе Хука-Дживса?
8. В чем отличия метода Розенброка и метода Хука-Дживса? Каковы условия применения метода Розенброка?
9. Дайте определение регулярного симплекса.
10. Как строится новый симплекс на основе базового?
11. В чем отличие процедур отражения и сжатия вовнутрь?
12. Какие векторы называются H-сопряженными?
13. Сформулируйте основную идею метода Пауэлла.
14. Как выбирается направление и параметр шага в методе наискорейшего спуска?
15. В чем отличие выбора направлений метода сопряженных градиентов и метода сопряженных направлений? Как выбирается параметр шага в методе Флетчера Ривса?
16. Как выбирается параметр шага в методе Поллака Райвера?
17. Какие направления поиска называются ньютоновскими?
18. В чем состоит основной принцип модификаций метода Ньютона?
19. Как строится релаксационная последовательность в методе Марквардта?
20. Какие методы называются квазиньютоновскими?
21. Как аппроксимируют обратную матрицу Гессе в методах Пирсона? Какая матрица выбирается в качестве начальной?
22. В чем отличие методов Пирсона и метода Дэвидона Флетчера Пауэлла?
23. В чем отличие стратегий линейного и нелинейного поиска в методах случайного поиска?

3.5. Контрольные вопросы к разделу 5 и 6

1. Допустимые направления и выделение ограничений
2. Необходимые условия Куна-Таккера.
3. Геометрическая интерпретация условий Куна-Таккера.

4. Седловые точки. Свойство седловых точек.
5. Функция Лагранжа.
6. Достаточное условие оптимальности.
7. Существование седловой точки для задач выпуклого программирования
8. Связь с условиями Куна-Таккера.
9. Принципы построения численных методов поиска условного экстремума.
10. Методы последовательной безусловной оптимизации
11. Методы возможных направлений.
12. Метод штрафов. Постановка задачи. Стратегия поиска.
13. Виды штрафных функций. Сходимость метода.
14. Метод барьерных функций. Постановка задачи. Стратегия поиска.
15. Виды барьерных функций. Сходимость метода.
16. Комбинированный метод штрафных функций. Постановка задачи. Стратегия поиска.
17. Метод множителей. Постановка задачи. Стратегия поиска.
18. Сходимость метода множителей.
19. Метод точных штрафных функций. Постановка задачи. Стратегия поиска.
20. Сходимость метода точных штрафных функций.
21. Метод проекции градиента. Постановка задачи. Стратегия поиска.
22. Сходимость метода проекции градиента.
23. Метод Зойтендейка. Постановка задачи. Стратегия поиска.

.

4. ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ

Белорусский национальный технический университет

УТВЕРЖДАЮ
Проректор по учебной
работе Белорусского
национального
технического университета
_____ А.Г. Баханович
24.11.2016.
Регистрационный № УД-М-ФИТР50-09/уч.

АЛГОРИТМИЧЕСКАЯ РЕАЛИЗАЦИЯ ЧИСЛЕННЫХ МЕТОДОВ

Учебная программа учреждения высшего образования
по учебной дисциплине для специальности
1-31 81 12 «Прикладной компьютерный анализ данных»

2016г.

Учебная программа составлена на основе образовательного стандарта ОСВО 1-1-31 81 12 2015

СОСТАВИТЕЛИ:

В.М. Волков, профессор кафедры «Системы автоматизированного проектирования» Белорусского национального технического университета, доктор физико-математических наук, доцент.

И.Л.Ковалева, доцент кафедры «Системы автоматизированного проектирования» Белорусского национального технического университета, кандидат технических наук, доцент.

РЕЦЕНЗЕНТЫ:

М.В.Игнатенко, доцент кафедры «Веб-технологии и компьютерное моделированиие» механико-математического факультета Белорусского государственного университета, кандидат физико-математических наук, доцент;

Ю.Б.Попова, доцент кафедры «Программное обеспечение вычислительной техники и автоматизированных систем» Белорусского национального технического университета, кандидат технических наук, доцент.

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой «Системы автоматизированного проектирования» Белорусского национального технического университета
(протокол № 12 от 27.06 2016 г.)

Заведующий кафедрой _____ А.В.Бородуля

Методической комиссией факультета информационных технологий и робототехники Белорусского национального технического университета (протокол № 10 от 30.06 2016 г.)

Председатель методической комиссии _____ С.В.Васильев

Научно-методическим советом Белорусского национального технического университета (протокол № 6 секции №1 от 31.08. 2016 г.)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Учебная программа по учебной дисциплине «Алгоритмическая реализация численных методов» предназначена для второй ступени обучения на факультета информационных технологий и робототехники Белорусского национального технического университета по специальности 1-31 81 12 «Прикладной компьютерный анализ данных».

Целью учебной дисциплины является углубленное изучение методов численного решения прикладных задач, их качественного анализа, а также приобретение навыков самостоятельной реализации численных методов на персональных компьютерах.

Основные задачи преподавания дисциплины «Алгоритмическая реализация численных методов» состоят в:

– формировании целостного представления о совокупности методов численного анализа, общих подходов к построению различных методов;

– приобретение и совершенствование навыков программной реализации численных методов и методов оптимизации;

– обучение технике решения прикладных задач в современных пакетах для моделирования процессов и инженерного анализа.

Изучение дисциплины базируется на знаниях, полученных из курсов высшей математики, физики, программирования, основ конструирования и деталей машин. Освоение навыков работы с современным ПО для моделирования процессов и инженерного анализа в качестве пользователя представляется важным с точки зрения развития умений практического использования математических знаний и одновременно является неотъемлемой составляющей в подготовке специалистов по разработке такого рода продукции.

В результате изучения дисциплины «Алгоритмическая реализация численных методов» магистрант должен:

знать:

– постановки задач для дифференциальных уравнений различных типов;

– основные численные методы решения дифференциальных уравнений;

- основные методы условной и безусловной оптимизации

уметь:

– переходить от реальной задачи к математической постановке;

– алгоритмически реализовывать основные численные методы решения дифференциальных уравнений и методы оптимизации;

владеть:

– навыками использования современных программных средств для компьютерной реализации численного решения дифференциальных уравнений и методов оптимизации.

Освоение данной учебной дисциплины обеспечивает формирование следующих компетенций:

АК-2. Методологические знания и исследовательские умения, обеспечивающие решение прикладных задач и инновационной деятельности.

СЛК-1. Учитывать социальные и нравственно-этические нормы в социально-профессиональной деятельности.

ПК-1. Квалифицированно использовать современные достижения по разработке и анализу математических моделей, методов компьютерного анализа данных и современные информационные технологии.

Согласно учебному плану для очной формы получения высшего образования на изучение учебной дисциплины отведено всего часов – 172, из них аудиторных 72 часа.

Распределение аудиторных часов по курсам, семестрам и видам занятий приведено в таблице 1.

Таблица 1

Очная форма получения высшего образования					
Курс	Семестр	Лекции, ч.	Лабораторные занятия, ч.	Практические занятия, ч.	Форма текущей аттестации
1	2	36	36		зачет

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Раздел I. Численные методы для обыкновенных дифференциальных уравнений

Тема 1.1. Задача Коши

Постановка задачи. Метод Эйлера. Явные и неявные схемы. Понятие устойчивости. Методы Рунге-Кутта. Адаптация шага сетки. Многошаговые методы. Жесткие системы. Метод Гира.

Тема 1.2. Краевые задачи

Постановка задачи. Метод стрельбы. Метод конечных разностей и конечных элементов для решения краевых задач. Спектральные методы.

Раздел 2. Численные методы для дифференциальных уравнений с частными производными

Тема 2.1. Метод конечных разностей

Нестационарное одномерное уравнение теплопроводности. Согласованность, устойчивость и сходимость. Эллиптические уравнения в прямоугольной области. Нелинейные уравнения. Методы переменных направлений и дробных шагов. Многосеточные методы.

Тема 2.2. Метод конечных элементов

Слабая формулировка дифференциальной задачи. Конечно-элементная дискретизация. Метод Галеркина. Смешанные производные и граничные условия. Дискретная конечно-элементная модель. Примеры прикладного программного обеспечения на основе метода конечных элементов.

Раздел 3. Методы линейного программирования.

Тема 3.1. Линейное программирование

Симплекс-метод. Двойственность. Параметрическое линейное программирование. Метод Кармаркара. Модели линейного программирования.

Тема 3.2. Целочисленное линейное программирование

Введение. Примеры задач целочисленного программирования. Методы решения задач целочисленного программирования.

Тема 3.3. Методы решения транспортных задач

Постановка задачи и стратегии решения. Методы нахождения начального плана перевозок. Метод потенциалов.

Раздел 4. Численные методы поиска безусловного экстремума

Тема 4.1. Основные методы безусловной оптимизации

Методы нулевого порядка. Методы первого порядка. Методы второго порядка (Метод Ньютона. Метод Ньютона-Рафсона. Метод Марквардта).

Раздел 5. Нелинейная оптимизация с ограничениями

Тема 5.1. Необходимые условия оптимальности

Допустимые направления и выделение ограничений. Необходимые условия Куна-Таккера. Геометрическая и физическая интерпретация.

Тема 5.2. Достаточные условия оптимальности

Седловые точки и функции Лагранжа. Существование седловой дочки для задач выпуклового программирования. Связь с условиями Куна-Таккера.

Раздел 6. Численные методы поиска условного экстремума

Тема 6.1. Методы последовательной безусловной минимизации

Метод штрафов. Метод барьерных функций. Метод множителей. Метод точных штрафных функций.

Тема 6.2. Методы возможных направлений

Метод проекции градиента. Метод Зойтендейка

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ

Номер раздела,	Название раздела, темы	Количество аудиторных часов					Форма Контроля знаний
		Лекции	Практические занятия	Лабораторные занятия	Иное		
1	2	3	4	5	7	8	9
	2 семестр						
1.	Численные методы для обыкновенных дифференциальных уравнений						
1.1	Задача Коши	4					
1.2	Краевые задачи	4					
2	Численные методы для дифференциальных уравнений с частными производными						
2.1	Метод конечных разностей	4		4			
2.2	Метод конечных элементов	4		6			
3.	Методы линейного программирования						
3.1	Линейное программирование	4		4			
3.2	Целочисленное линейное программирование	4		4			
3.3	Методы решения транспортных задач	4		4			
4	Численные методы поиска безусловного экстремума						
4.1	Основные методы безусловной оптимизации	2		6			
5	Нелинейная оптимизация с ограничениями						
5.1	Необходимые условия оптимальности	1		4			

5.2	Достаточные условия оптимальности	1					
6	Численные методы поиска условного экстремума						
6.1	Методы последовательной безусловной минимизации	2		4			
6.2	Методы возможных направлений	2					
	Итого за семестр	36		36			зачет
	Всего аудиторных часов	72					

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Список литературы

Основная литература

1. Ануфриев И. Е. Применение PDE Toolbox при изучении некоторых разделов вычислительной математики // Труды III научной конференции «Проектирование инженерных и научных приложений в среде MATLAB Санкт-Петербург, 2007 С. 42-56
2. Дьяконов В. П. MATLAB 6/6.1/6.5+ Simulink 4/5 в математике и моделировании. – Издательский дом" Солон-Пресс", 2009.
3. Колемаев В. А., Малыхин В. И., Соловьев В. И. и др. Математические методы и модели исследования операций. М.: ЮНИТИ-ДАНА, 2008
4. Методы оптимизации в примерах и задачах: Учеб.пособие / А.В. Пантелеев, Т.А. Летова – 3-е изд., исправл. – М.: Высш.шк., 2010 – 544 с., ил
5. Оптимизационные расчеты на основе ANSYS – методические указания к выполнению лабораторных, курсовых и дипломных работ / сост. В.В. Напрасников, Ю.В.Напрасникова, А.В. Бородуля, А.Н.Соловьев – Минск: БНТУ, 2012 – 37 с., ил.
6. Особенности расчета конструкций с шарнирами в ANSYS– методические указания/ сост. В.В. Напрасников, Ю.В.Напрасникова, С.В.Красновская, А.Н.Соловьев – Минск: БНТУ, 2014 – 32 с., ил.
7. Писарук Н. Н. Модели и методы смешанно-целочисленного программирования. — Мн.: Изд-во БГУ, 2010.
8. Численные методы: методические указания и индивидуальные задания для студентов –заочников / Белорусский национальный технический университет, кафедра «Высшая математика №1»; сост.О.Р.Габасова, А.В.Грекова и др. – Минск: БНТУ, 2009 – 116 с., ил.
9. Численный анализ и оптимизация /В.М.Волков, О.Л.Зубко, И.Н.Катковская, И.Л.Ковалева, В.Г.Кротов, П.Лима. – Минск: РУП «Белгослес», 2017-207 с.

10. Численные методы для обыкновенных дифференциальных уравнений [Электронный ресурс]: учебно-методическое пособие для магистрантов специальности 1-31 81 12 «Прикладной компьютерный анализ данных»/ В.М.Волков, И.Л.Ковалева – БНТУ, 2016.

11. Хэмди А. Таха Введение в исследование операций, 8-е издание.: Пер. с англ. – М.: Вильямс, 2007 – 912 с.

Дополнительная литература

12. matlab.exponenta.ru/index.php

13. http://www.mathworks.com/products/parallel-computing/?s_tid=hp_fp_list

14. http://www.mathworks.com/mathematical-modeling/

15. Компьютерные технологии решения инженерных задач в MATLAB [Электронный курс]: учебно-методическое пособие/ Краков М.С. и др. – БНТУ, 2012.

16. Сидорик В.В. Практикум по моделированию в среде MATLAB – учебно-методическое пособие – Электрон.дан.- БНТУ, 2012

17. Васильев В. В., Симак Л. А., Рыбникова А. М. Математическое и компьютерное моделирование процессов и систем в среде MATLAB/SIMULINK //Учебное пособие для студентов и аспирантов. Киев: НАН. Украины. – 2008.

Средства диагностики результатов учебной деятельности

Оценка уровня знаний магистранта производится по десятибалльной шкале в соответствии с критериями, утвержденными Министерством образования Республики Беларусь.

Для оценки достижений магистранта рекомендуется использовать следующий диагностический инструментарий:

- устный и письменный опрос во время лабораторных занятий;

– защита выполненных на лабораторных занятиях и в рамках самостоятельной работы индивидуальных заданий;
– собеседование при проведении индивидуальных и групповых консультаций;
– сдача зачета по дисциплине.

Методические рекомендации по организации и выполнению самостоятельной работы магистрантов

При изучении дисциплины рекомендуется использовать следующие формы самостоятельной работы:

– решение индивидуальных задач в аудитории во время проведения лабораторных занятий под контролем преподавателя в соответствии с расписанием;
– подготовка рефератов по индивидуальным темам, в том числе с использованием патентных материалов.