Билет 2.

1. Численное решение уравнений в частных производных методом конечных разностей. Задачи с начальными условиями и граничные задачи.

Большая часть задач компьютерного моделирования и анализа физических, химических, биологических и многих других систем описываются уравнениями и системами уравнений с частными производными. Методы решения этих задач хорошо развиты и успешно используются в различных областях. В большинстве математических книг уравнения в частных производных делятся на основе их характеристик на три группы: гиперболические, параболические и эллиптические. Примером уравнения гиперболического типа может служить уравнение волны:

$$\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2}$$

(1.1) , где v – скорость распространения волны. Примером уравнения параболического типа является уравнение диффузии

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}(D\frac{\partial u}{\partial x})$$

(1.2) Уравнение Пуассона – уравнение эллиптического типа

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y)$$

(1.3) Здесь ρ(x,y) - плотность заряда в среде. Мы будем рассматривать численные решения уравнений в частных производных, и с вычислительной точки зрения приведенная выше классификация не очень важна. Для первого и второго типа уравнений ставится задача с начальными условиями или задача Коши, При решении такой задачи заданы значения функции u или ее производных в начальный момент времени t_0 для всех точек x, а уравнение описывает изменение решения со временем. В этом случае цель численного решения – проследить временные изменения с требуемой точностью. Уравнение третьего типа описывает статическую функцию u(x,y), которая удовлетворяет уравнению внутри некоторой интересующей нас области и удовлетворяет заданным условиям на границе области. Такие задачи называются задачами с граничными условиями. В этом случае – цель численного решения – найти правильное решение во всей области. С вычислительной точки зрения важнее разделять уравнения на уравнения, которые описывают задачу с начальными условиями (1 и 2 типа с решениями меняющимися во времени) и на уравнения, которые описывают граничную задачу (3 тип – статические решения).*Задачи с начальными условиями.* При постановке задачи этого типа должны быть даны ответы на следующие вопросы: Какие зависимые переменные изменяются со временем? Каким временным уравнением описывается каждая переменная? Обычно в правой части каждого уравнения присутствуют больше, чем одна зависимая переменная. Какой наивысший порядок производной по времени во временном уравнении для каждой переменной? Если возможно, эту производную по времени надо поместить одну в левую часть уравнения. Для решения задачи необходимо задать значения всех необходимых производных по времени (до производных наивысшего порядка, присутствующих в уравнении) в начальный момент. Каким граничным условиям должно удовлетворять решение? Это могут быть условия Дирихле, задающие значение функции, зависящей от времени в граничных точках, условия Неймана, задающие значения производной по нормали к границе, условия, соответствующие бегущей волне. Самое важное требование к алгоритму, используемому для решения этих задач – *стабильность. Граничные задачи.* При постановке задачи этого типа должны быть даны ответы на следующие вопросы:

Каковы переменные? Каким уравнениям удовлетворяет решение внутри интересующей нас области? Каким уравнениям удовлетворяет решение в точках на границе? (Здесь также могут быть условия Дирихле или Неймана, но могут быть и гораздо более сложные условия.) В отличие от задач с начальными условиями стабильность алгоритма достигается достаточно легко. Самое важное требование к алгоритму, используемому для решения этих задач *–эффективность.* Все условия в граничных задачах должны быть выполнены «одновременно», поэтому решение таких задач сводится к решению системы большого количества алгебраических уравнений. Если эти уравнения оказываются нелинейными, они линеаризуются или решаются методом итераций, т.е. не теряя общности можно считать, что задача сводится к решению специальной системы, содержащей большое количество линейных уравнений. Рассмотрим решение уравнения (3) конечно-разностным методом. Мы представляем функцию u(x,y) значениями в узлах прямоугольной сетки $x_j = x_0 + j\Delta$ $j=0,1,...,J$ $y_l = y_0 + l\Delta$ $l=0,1,...,L$ (1.4) где Δ- шаг сетки (одинаковый и по x и по y). Вторые производные можно заменить следующими выражениями

$$\frac{u_{j+1,l} - 2u_{j,l} + u_{j-1,l}}{\Delta^2} + \frac{u_{j,l+1} - 2u_{j,l} + u_{j,l-1}}{\Delta^2} = \rho_{j,l} \quad (1.5)$$ или

$$u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 \rho_{j,l}$$

(1.6) Чтобы записать систему линейных уравнений в матричной форме перенумеруем точки двумерной сетки одномерной последовательностью I =j*(L+1)+l для j=0,1,...,J, l=0,1,...,L (1.7) Уравнение (6) принимает вид $$u_{i+L+1} + u_{i-(L+1)} + u_{i+1} + u_{i-1} - 4u_i = \Delta^2 \rho_i$$ (1.8) оно справедливо для внутренних точек области j=1,2,...,J-1; l=1,2,...L-1 Граничные точки области, в которых заданы функции или производные, приведены ниже

j=0 [i.e., i =0 ,...,L]
j=J [i.e., i =J (L +1),...,J (L +1)+L] (1.9)
l=0 [i.e., i =0 ,L +1 ,...,J (L +1)]
l=L [i.e., i =L,L +1+L,...,J (L +1)+L]

всю эту информацию (о значениях на границе) перенесем в правую часть уравнения (8) и уравнение примет вид: **A*u=b** (1.10) Общее эллиптическое уравнение второго порядка при построении разностной схемы приводит к матрице аналогичной описанной выше.

Существуют три различных подхода к решению уравнения (1.10): Методы релаксации, «быстрые методы» (методы Фурье) и прямые матричные методы. Методы релаксации явно используют структуру разреженной матрицы **A.** Матрица делится на две части **A=E-F** (1.11) Где E легко инвертируется, а F – то что осталось, тогда: **E*u=F*u+b** (1.12) При использовании метода релаксации сначала выбирается начальное приближение, а затем итеративно вычисляется следующее $E \cdot u^{(r)} = F \cdot u^{(r-1)} + b$ (1.13)

E легко инвертируемо, так что итерации быстрые. Так называемые быстрые методы применимы только для специального класса уравнений: с постоянными коэффициентами, или, в более общем случае к уравнениям с разделяющимися переменными, Кроме того, границы области должны совпадать с координатными линиями. Заметим однако, что существуют многосеточные методы релаксации, которые могут быть быстрее «быстрых» методов. При помощи матричных методов прямо решают уравнение $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ (1.14)

Конкретное решение очень сильно зависит вида матрицы. При решении надо учитывать разреженность матрицы, в противном случае задача осложняется большими размерами матрицы, так при сетке 100*100 надо найти 10000 $u_{j,l}$, что приводит к матрице **A** размером 10000*10000. Если A симметрична и положительно определена, как, например в случае эллиптического уравнения, то можно использовать алгоритм сопряженного градиента. Кроме метода конечных разностей существует множество других методов решения уравнений с частными производными: метод конечных элементов, Монте-Карло, вариационные методы и другие.

2. Преобразование результатов проектирования в программный код.

Написание кода на объектно-ориентированном языке программирования – конечная цель проектирования. Она реализуется на основе созданных на этапе проектирования артефактов: концептуальной модели, диаграмм взаимодействия (в частности диаграмм кооперации), диаграммы классов. На основе диаграмм классов создаются определения классов, если на диаграмме кооперации системной операции передается сообщение create, то надо добавлять конструктор. На диаграмме классов атрибуты-ссылки представлены ассоциациями и связанным ними направлением перемещений. Атрибуты – ссылки классов зачастую косвенно присутствуют, а не явно определяются на диаграмме классов. На основе диаграмм кооперации создаются методы классов. Каждое сообщение из последовательности внутри метода, как показано на диаграмме кооперации преобразуется в оператор метода на объектно-ориентированном языке. Классы нужно реализовывать (и, в идеале, полностью тестировать) от минимально связанных с другими классами до максимально связанных. Процесс преобразования объектно-ориентированных диаграмм классов в их определения и диаграмм кооперации в методы является относительно простым. Общая архитектура и основные решения должны быть приняты до перехода к стадии кодирования. Преобразование результатов проектирования в программный код - это последний этап при разработке объектно-ориентированных компьютерных приложений. Важно отметить, что разработка приложений – процесс итерационный, при котором проектирование и кодирование могут повторяться много раз.