МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ

Методические указания по выполнению лабораторных работ

Направление подготовки 03.03.02 Физика
Направленность (профиль)
«Фундаментальная и прикладная физика»

Ставрополь
2026

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность учебной дисциплины «Компьютерное моделирование физических процессов» обусловлена необходимостью формирования у будущих бакалавров общепрофессиональных компетенций, практических умений на которых базируется профессиональная деятельность в физике. В современных условиях важнейшим элементом подготовки бакалавров в области физики является формирование у них самостоятельности и культуры научного мышления, способности вести научную и профессиональную деятельность, навыков постановки целей, задач при решении проблемных ситуаций.

Целью лабораторных занятий является формировании у студентов, обучающихся по направлению подготовки 03.03.02 Физика компетенции: ПК-3 и ПК–5

Основные задачи дисциплины:

Задачей настоящего курса является ознакомление студентов с особенностями основных методов физических исследований и технологиями математического и компьютерного моделирования, связанного с основными направлениями современной физики, в выработке практических навыков решения физических проблем

В результате изучения дисциплины у обучающихся формируются следующие компетенции (в соответствии с образовательным стандартом):

ИД-1$_{ПК-3}$ понимает стратегию и тактику проектной деятельности как целенаправленной, системной, профессиональной деятельности

ИД-2$_{ПК-3}$умеет применять приобретенные знания, умения и навыки в своей проектной деятельности;

ИД-1$_{ПК-5}$ ориентируется в современных тенденциях развития цифровых технологий, выбирает технологии или программные средства для решения поставленных задач.

ИД-2$_{ПК-5}$ умеет обрабатывать и анализировать полученные данные с помощью современных информационных технологий;

ИД-3$_{ПК-5}$ обладает готовностью к участию в подготовке проектной документации, в том числе с использованием средств автоматизированного проектирования и вычислительных программных комплексов, современного программного обеспечения, в том числе текстовых редакторов и графических программы;

ИД-4$_{ПК-5}$ владеет современным программным обеспечением, в том числе текстовыми редакторами и графическими программами, средствами подготовки обзоров, отзывов, отчетов, заключений

Цель данного практикума – описание последовательности, основного содержания, методики выполнения практических занятий по учебному курсу и (или) самостоятельной подготовки к ним. Занятия проводятся в форме лабораторных работ.

ТЕМА 1. МОДЕЛИРОВАНИЕ ОТНОСИТЕЛЬНЫХ ДВИЖЕНИЙ В КЛАССИЧЕСКОЙ МЕХАНИКЕ

Построение орбиты луны в гелиоцентрической системе отсчета

С точки зрения кинематического подхода эта задача соответствует первой постановке задачи об относительном движении. В гелиоцентрической системе отсчета (система K) Земля движется по окружности радиуса $R_1 = 1, 496 \cdot 108$ км (период обращения $T_1 = 3, 156 \cdot 107$ с). Луна в свою очередь движется вокруг Земли (система K) по окружности радиуса $R_2 = 3, 844 \cdot 105$ км (период обращения T_2=2, 36 · 106 с). Как известно [1, 2], при движении материальной точки по окружности радиуса R с постоянной угловой скоростью w координаты радиуса-вектора, проведенного из начала координат к текущему положению точки, меняются по закону

$$\vec{\boldsymbol{R}}(\boldsymbol{t}) = \begin{pmatrix} \boldsymbol{R}\cos(\boldsymbol{\omega t} + \boldsymbol{\varphi_0}) \\ \boldsymbol{R}\sin(\boldsymbol{\omega t} + \boldsymbol{\varphi_0}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}\cos\left(\dfrac{2\pi}{T}\boldsymbol{t} + \boldsymbol{\varphi_0}\right) \\ \boldsymbol{R}\sin\left(\dfrac{2\pi}{T}\boldsymbol{t} + \boldsymbol{\varphi_0}\right) \end{pmatrix}$$

где φ_0 – начальная фаза, характеризующая положение частицы в момент времени t = 0, которую в дальнейшем мы будем полагать равной нулю. Заменяя в (1.5) R на R_1, R_2 и подставляя в уравнение движения получаем зависимость радиуса-вектора Луны в гелиоцентрической системе координат от времени:

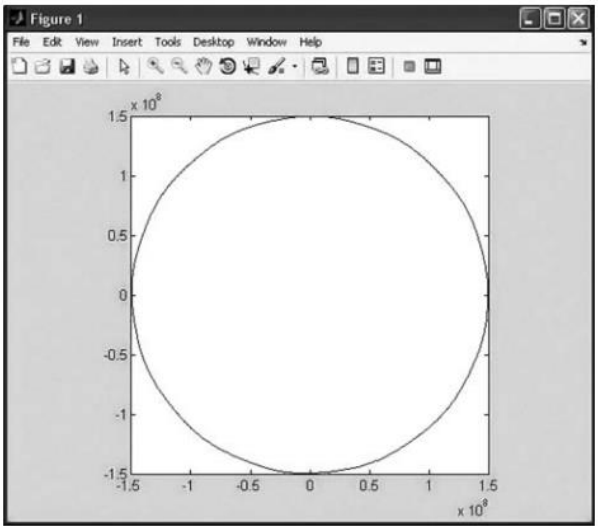
$$\vec{\boldsymbol{r}}(\boldsymbol{t}) = \begin{pmatrix} \boldsymbol{x}(\boldsymbol{t}) \\ \boldsymbol{y}(\boldsymbol{t}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{R_2}\cos\left(\dfrac{2\pi}{T_2}\boldsymbol{t}\right) + \boldsymbol{R_1}\cos\left(\dfrac{2\pi}{T_1}\boldsymbol{t}\right) \\ \boldsymbol{R_2}\sin\left(\dfrac{2\pi}{T_2}\boldsymbol{t}\right) + \boldsymbol{R_1}\sin\left(\dfrac{2\pi}{T_1}\boldsymbol{t}\right) \end{pmatrix}$$

Это выражение задает орбиту Луны

Для построения искомой орбиты в MATLAB в режиме непосредственного вычисления необходимо передать пакету следующую последовательность команд, сохраненную в файле Glava1_1.m. (В MATLAB часть строки, расположенная после знака %, является комментарием и при самостоятельном вводе команд может быть опущена.)

```
>>  R1=1.496*10^8; % задание численного значения
                   % радиуса орбиты Земли
>>  T1=3.156*10^7; % задание численного значения периода
                   % обращения Земли вокруг Солнца
>>  R2=3.844*10^5; % задание численного значения
                   % радиуса орбиты Луны
>>  T2=2.360*10^6; % задание численного значения
                   % периода обращения Земли  вокруг Земли
>>  t=0:T1/1000:T1; % задание дискретной переменной,
                    % изменяющейся от 0 до T1 с шагом T1/1000
>>  Xz=R1*cos(2*pi*t/T1); % вычисление x-й координаты
                          % радиуса-вектора Земли
>>  Yz=R1*sin(2*pi*t/T1); % вычисление y-й координаты
                          % радиуса-вектора Земли
>>  Xm=R2*cos(2*pi*t/T2); % вычисление x-й координаты
                          % радиуса-вектора Луны
                          % в системе координат, связанной с Землей
>>  Ym=R2*sin(2*pi*t/T2); % вычисление y-й координаты
                          % радиуса-вектора Луны
                          % в системе координат, связанной с Землей
>>  Xotn=Xz+Xm; % вычисление x-й координаты
                % радиуса-вектора Луны
                % в гелиоцентрической системе координат
>>  Yotn=Yz+Ym; % вычисление y-й координаты
                % радиуса-вектора Луны
                % в гелиоцентрической системе координат
>>  plot(Xotn,Yotn); % визуализация орбиты Луны в гелиоцентрической
                     % системе координат (рис. (1.2))
```

```
>>  h = gca; % загрузка дискриптора графического окна
>>  set(h,'DataAspectRatioMode','manual'); % задание режима
                                           % отображения
                                           % пропорциональных
                                           % отрезков по обеим
                                           % координатным осям
```

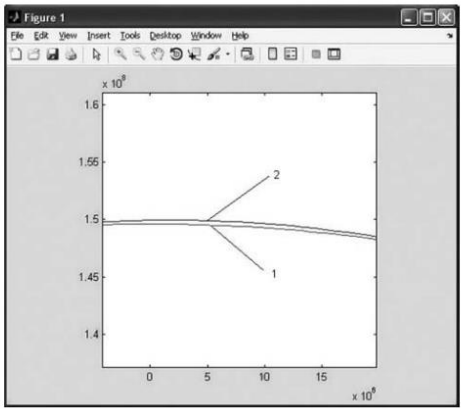


Для отображения на одном графике орбиты Земли и орбиты Луны в гелиоцентрической системе координат следует вместо команды

>> plot(Xotn,Yont)

ввести команду

>> plot(Xotn,Yont,Xz,Yz)



Фрагмент траекторий Земли и Луны в гелиоцентрической системе координат представлен на рисунке

Для получения информации о всех активных переменных, находящихся в памяти компьютера (рабочей области) в данный момент времени, используется команда whos.

Отметим, что постоянные в MATLAB трактуются как матрицы размерности 1 × 1.

Для сохранения значений переменных, находящихся в рабочей области, используется

команда

>> save имя_файла

или функция

>> save (имя_файла)

Команда save имеет несколько различных форм:

>> save имя_файла – записываются все переменные рабочей области в файл «имя_файла» с расширением .m;

save имя_файла X – записывается только значение переменной X;

save (имя_файла) X Y Z –записываются только значения переменных X, Y и Z.

После записи команды save также можно указывать ключи, уточняющие формат записи файлов:

-mat — двоичный MAT-формат, используемый по умолчанию;

-ascii — ASCII-формат единичной точности (8 цифр);

-ascii -double — ASCII-формат двойной точности (16 цифр);

-ascii -double -tabs — ASCII-формат двойной точности с разделителем и метками табуляции;

V4 — запись MAT-файла в стандарте версии MATLAB 4.0;

-append — добавление в существующий MAT-файл.

Для загрузки ранее проведенного сеанса работы с MATLAB используется команда

>> load имя_файла

или функция

>> load(имя_файла)

Приведенный выше протокол команд можно сохранить в виде файла на диске для последующего анализа решения или использования, как основы программы сценария, используемой при решении подобных задач, командой

>> diary имя_файла

или функцией

>> diary(имя_файла)

Например, для записи в файл Moon.m протокола приведенных выше команд необходимо ввести следующую команду:

>> diary Moon.m % открытие файла для сохранения протокола

Далее каждая последовательно выполненная команда будет заноситься в файл Moon.m. Для приостановки записи выполняемых команд в файл используется команда

diary off

Отметим, что данная команда также записывается в файл Moon.m. Начиная с версии MATLAB 6.0 и выше, данный набор команд можно выполнить автоматически, набрав в командной строке имя файла Moon и нажав клавишу <Enter>.

Типовые задания

Задача 1. Определите предельное значение радиуса орбиты Луны, при котором не происходит появление петель. Как выглядит орбита Луны в гелиоцентрической системе координат в этом случае?

Задача 2. Зафиксируйте радиус орбиты Луны и постройте орбиту Луны при различных значениях периода обращения. Что можно сказать о размере петель в этом случае?

ТЕМА 2. ФИЗИЧЕСКИЕ ПРОЦЕССЫ, ОПИСЫВАЕМЫЕ ДИФФЕРЕНЦИАЛЬНЫМИ УРАВНЕНИЯМИ ПЕРВОГО ПОРЯДКА

Приступая к разработке программы вне зависимости от использованного языка программирования, необходимо разбить всю задачу на последовательность независимых заданий, соответствующих алгоритму, описанному в предыдущем разделе. Программа должна состоять из следующих блоков:

1. Задание начальных условий.
2. Задание функции $\boldsymbol{f(x, y(x))}$.
3. Задание отрезка, на котором ищется решение, и шага интегрирования. (Отметим, что на

практике оказывается более удобным задавать не шаг интегрирования, а количество интервалов, на которые разбивается отрезок интегрирования,
а затем вычислять значение шага.)

4. Вычисление координат точек, в которых ищется решение дифференциального уравнения.

5. Решение конечно-разностного уравнения методом Эйлера.

6. Вывод результатов.

Для примера рассмотрим решение задачи Коши для ДУ

$$y' = -y^2 + x$$

с начальным условием $y(0) = 1$

Для решения поставленной задачи в MATLAB потребуется создание двух файлов, называемых m-файлами, потому что имена этих файлов имеют вид <Имя>.m.

Отметим, что большая часть времени пользователя MATLAB состоит в создании, редактировании и выполнении m-файлов. В MATLAB существуют два вида файлов: файлы-программы и файлы-функции. Файлы программы состоят из последовательности обычных операторов MATLAB. Примером программы является, например, файл сценария, содержащий последовательность команд, выполненных в предыдущем разделе при моделировании относительных движений. Переменные, используемые в программе, являются глобальными — они изменяют значения переменных с аналогичным названием в рабочей области текущей сессии. Файлы-функции содержат описания специфических функций, созданных пользователем для решения конкретной задачи. Они дают возможность пользователю фактически расширить возможности MATLAB, поскольку имеют тот же статус, что и другие функции пакета.

Любой модуль, содержащий функцию, возвращающую один выходной параметр, имеет следующую структуру

```
function var = name_function(список_параметров)
% Комментрии
Блок команд, реализующих вычисление функции
var = выражение;  % данная строчка вводится,
                  % если функция возвращает
                  % результат вычислений
```

Свойства m-файла, содержащего m-функцию:

1. m-файл начинается с объявления типа function, после которого указывается имя переменной, являющейся параметром, возвращаемым функцией, имя самой функции и список формальных параметров, передаваемых в функцию при ее вызове.

2. Если последняя строка функции имеет вид var=выражение, то в качестве значения

функции возвращается значение выражения, занесенного в переменную var.

3. Все переменные, используемые в теле функции, являются локальными переменными, то есть их определение действует только внутри данной функции.

4. Для использования глобальных переменных их список приводится в строке, предваряющей блок проведения вычислений:

```
function var = name_function(список_параметров)
% Комментрии
global var1, var2, var3 % объявление глобальных переменных
Блок команд, реализующих вычисление функции
var = выражение;  % данная строчка вводится,
                  % если функция возвращает
                  % результат вычислений
```

5. Функция является самостоятельным программным модулем, взаимодействующим с другими программными модулями через входные, выходные и глобальные переменные.

6. В файле функции допускаются комментарии, начинающиеся символом %.

7. При вызове файла-функции сначала происходит его компиляция, а затем исполнение. Машинный код m-функции хранится в рабочей области MATLAB.

8. Обращение к функции осуществляется указанием ее имени и значений переменных, перечисленных в списке формальных параметров:

```
имя_переменной = name_function(список_параметров)
name_function(список_параметров)
```

Любой модуль, содержащий функцию, возвращающую несколько выходных параметров, имеет следующую структуру:

```
function [var1,var2,...] = name_function(список_параметров)
% комментарии
Блок команд, реализующих вычисление функции
var1 = выражение1;
var2 = выражение2;
var3 = выражение3;
... ... ... ... ...
```

Обращение к функции, возвращающей несколько выходных параметров:

```
[var1,var2,...] = name_function(список_параметров)
```

После этого переменные var1,var2,... становятся определенными и их можно использовать в последующих математических выражениях.

При обращении к функции, возвращающей несколько выходных параметров, в виде: имя_переменной=name_function(список_параметров) в переменную

имя_переменной возвращается только значение переменной var1.

Создадим, следуя перечисленным выше правилам, функции, позволяющие найти численные решения задачи Коши для обыкновенного ДУ первого порядка. Первый из создаваемых нами файлов будет содержать функцию, стоящую в правой части уравнения, второй — реализацию вычислительной схемы метода Эйлера. Для создания m-файла необходимо запустить встроенный в MATLAB текстовый редактор, выбрав последовательно следующие пункты меню: File → New → M-File, приемы редактирования текста в котором аналогичны любому редактору текстов, работающему под управлением операционной системы Windows. Затем набрать во вновь создаваемом m-файле следующий текст:

```
function F = f(x,y)
% f(x,y) - функция, стоящая в правой части
% дифференциального уравнения (2.8)
F = x-y^2;
```

и сохранить его на жестком диске под именем f.m.

Далее аналогичным образом создаем файл, содержащий следующие команды, реализующие метод Эйлера:

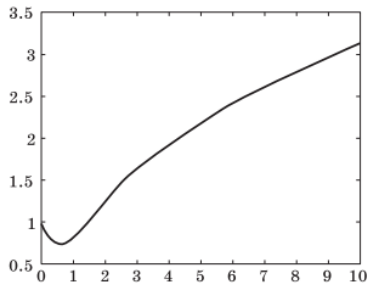
```
function [X,Y]=Euler(y0,x0,x1,N)
% функция, возвращающая численные
% решения дифференциального
% уравнения первого порядка
% методом Эйлера
dx=(x1-x0)/N; % вычисление шага
              % интегрирования
% задание начальных условий
x(1)=x0;
y(1)=y0;
% реализация вычислительной схемы
% метода Эйлера
for i=1:N
   x(i+1)=x(1)+dx*i;
   y(i+1)=y(i)+dx*f(x(i),y(i));
end;
% возвращение результатов вычислений
X=x;
Y=y;
```

и сохраняем его на диске под именем Euler.m. Функция Euler возвращает два вектора: X, Y — векторы, содержащие значения координат узлов сетки, на которой ищется решение ДУ, и значения решения ДУ в данных узлах соответственно.

После создания файлов f.m и Euler.m для получения решения ДУ и его визуализации следует в командном окне пакета ввести следующие команды, сохраненные нами в файле Glava2_1.m

```
% задание начальных условий
>> x0 = 0;
>> y0= 1;
>> x1 = 10; % задание правой
            % границы отрезка
>> N = 100; % задание числа узлов
            % координатной сетки
>> [x,y]=Euler(y0,x0,x1,N); % вычисление
                            % численного решения
                            % задачи Коши
                            % уравнения (2.8)
>> plot(x,y) % визуализация численного
            % решения задачи Коши
            % уравнения (2.8)
```

Отметим, что приведенную выше последовательность команд можно сначала ввести, для добавления новой строки в командном окне MATLAB одновременно нажимая клавиши Shift и Enter, и затем для выполнения всей последовательности команд нажать клавишу Enter.



Искомое численное решение дифференциального уравнения представлено на
рисунке

Теперь, когда создана программа для численного решения ДУ первого порядка, реализующая метод Эйлера, необходимо обсудить вопрос о точности получаемого решения. Так как мы заменили дифференциальное уравнение его разностным аналогом, то естественно ожидать некоторое отличие численного решения от «истинного» решения ДУ.

Типовые задания

Задача 1. Постройте на одном чертеже аппроксимирующие функции, коэффициенты которых вычислены каждым из трех выше перечисленных методов. Сравните сумму квадратов отклонений между табличными значениями и значениями аппроксимирующих функций, полученные при использовании функций
polyfit и lsqcurvefit.

Задача 2. Предположим, что начальная температура кофе 90◦C, однако начинать пить кофе можно, когда температура опустится ниже 60◦C. Предположим, что вы можете добавить в кофе молоко, которое уменьшает температуру кофе на 5◦C. Для того чтобы охладить кофе до нужной температуры вы можете поступить двумя способами: 1) добавить молоко при температуре кофе 90◦C; 2) добавить молоко при температуре кофе

75◦C. В каком случае охлаждение кофе до 60◦C произойдет быстрее? (Ответ необходимо получить, выполнив численные расчеты.)

ТЕМА 3. ДИНАМИКА МАТЕРИАЛЬНОЙ ТОЧКИ

Движение тел в гравитационном поле без учета трения

Рассмотрим задачу об одномерном движении материальной точки с постоянной массой в гравитационном поле Земли. В соответствии с законом всемирного тяготения сила, действующая на тело массы m, равна

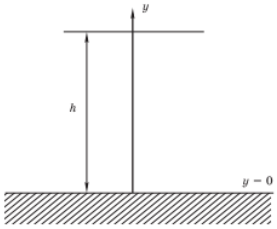
$$\boldsymbol{F = \frac{\gamma M m}{(R + y)^2} = \frac{g m}{\left(1 + \frac{1}{R}\right)^2}}$$

где y – расстояние от поверхности Земли, R – радиус Земли, g – гравитационная постоянная, M – масса Земли, $\boldsymbol{g = \gamma M / R_2}$. При этом на первом шаге будем считать силу тяжести постоянной, что справедливо, для случая, когда $\boldsymbol{\gamma / R \ll 1}$

Выберем систему отсчета с положительным направлением координатной оси вверх.

Уравнение движения в выбранной системе отсчета имеет вид:

$$\boldsymbol{\frac{d^2 y}{dt_2} = -g}$$



Данное уравнение является ДУ второго порядка, для которого задача Коши ставится заданием начальных условий: значений координаты $\boldsymbol{y(0) = y_0}$и скорости $\boldsymbol{v}$(0)=y′(0) $\boldsymbol{v_0}$. Аналитический ответ может быть получен последовательным интегрированием

$$\boldsymbol{v(t) = v_0 - g \cdot t}$$

$$\boldsymbol{y(t) = y_0 + v_0 \cdot t + \frac{1}{2} g \cdot t^2}$$

Рассмотрим последовательность действий, реализующих решение этой задачи в MATLAB с начальными условиями $\boldsymbol{y(0) = 10}$, $\boldsymbol{y'(0) = 1}$.

Поскольку все описанные выше решатели работают с системами ОДУ, приведенными к форме Коши, необходимо привести системе ОДУ второго порядка к системе обыкновенных дифференциальных уравнений первого порядка, введя новую переменную

$$\boldsymbol{\frac{dy}{dt} = z}$$

Тогда:

$$\frac{dz}{dt} = -g$$

Последние два уравнения образующие систему ОДУ первого порядка, как очевидно, можно записать в матричном виде:

$$\frac{d}{dt}\begin{pmatrix} y \\ z \end{pmatrix} = \begin{pmatrix} z \\ -g \end{pmatrix}$$

Запись системы в таком виде позволяет создать m-файл, содержащий описание вектора-функции, стоящей в правой части

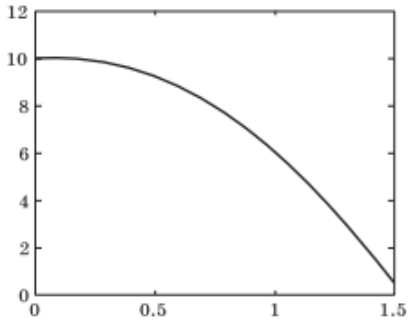
```
function dy=Angle(t,y)
  global g  % объявление глобальной переменной
  dy=zeros(2,1); % задание вектора-столбца размерности 2x1
  % задание координат вектора-функции,
  % стоящей в правой части (3.11)
  dy(1)=y(2);
  dy(2)=-g;
```

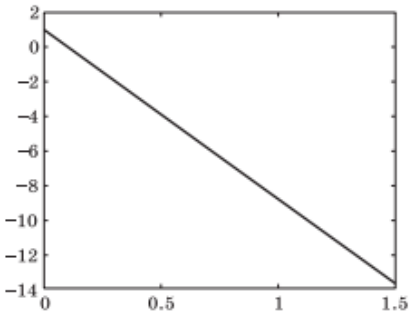
и сохранить его на диске под именем Angle.m.

Далее необходимо выполнить в командном окне следующую последовательность команд, сохраненную нами в файле Glava3_1.m:

```
% листинг файла Glava3_1.m
global g % объявление глобальной переменной
g=9.8; % задание значения
       % ускорения свободного падения
% задание начальных условий
y0=10;
v0=1;
% решение системы ОДУ (3.11)
[t,y]=ode45('Angle',[0:0.01:1.5],[y0 v0]);
figure(1); plot(t,y(:,1)); % построение зависимости x=x(t)
                           % в графическом окне No 1
figure(2); plot(t,y(:,2)); % построение зависимости y=y'(t)
                           % в графическом окне No 2
```

После выполнения приведенной выше последовательности команд на экране монитора появляются два независимых стандартных графических окна. В окне с названием Figure № 1 отображается зависимость координаты от времени, в окне с названием Figure № 2 отображается зависимость скорости от времени





Типовые задания

Задача 1. Проведите анализ полученных численных решений системы ДУ, описывающей движение материальной точки в поле тяжести Земли:

1. Сравните зависимости скорости и координаты от времени, с аналогичными зависимостями, полученными численным решением. Какова точность численного решения?

2. Постройте зависимость скорости от координаты.

3. Постройте зависимость пройденного пути материальной точки от времени.

4. Постройте зависимости кинетической и потенциальной энергий от времени.

5. Одним из фундаментальных законов физики является закон сохранения энергии, в соответствии с которым полная энергия материальной

точки, движущейся в гравитационном поле Земли, должна оставаться

постоянной. Проверьте полеченное численное решение ДУ на соответствие этому закону. С какой точностью выполняется закон сохранения

энергии и чем можно объяснить обнаруженные расхождения?

Задача 2. Шарик, имеющий небольшой радиус (по сравнению с чем?), падает с высоты h на упруго отражающую поверхность.

1. Разработайте программу, позволяющую исследовать движение шарика, падающего с заданной высоты на отражающую поверхность, предполагая, что удар является абсолютно упругим.
2. Постройте зависимости координаты, скорости, перемещения от времени и скорости от координаты.
3. Исследуйте движение шарика при неабсолютно упругом ударе

ТЕМА 4. ЗАДАЧА КЕПЛЕРА

Численное моделирование орбиты

$$\frac{d^2X}{d\tau^2} = -\frac{4\pi^2}{(X^2+Y^2)^{3/2}}X$$
$$\frac{d^2Y}{d\tau^2} = -\frac{4\pi^2}{(X^2+Y^2)^{3/2}}Y$$

Предваряя нахождение численного решения системы уравнений, приведем ее к эквивалентной системе уравнений первого порядка, выполнив замену переменных $X \to z_1$; $X^{/} \to z_2$; $Y \to z_3$; $Y^{/} \to z_4$

$$\frac{dz_1}{d\tau} = z_2$$

$$\frac{dz_2}{d\tau} = -\frac{4\pi^2}{(z_1^2 + z_3^2)^{3/2}} z_1$$

$$\frac{dz_3}{d\tau} = z_4$$

$$\frac{dz_4}{d\tau} = -\frac{4\pi^2}{(z_1^2 + z_3^2)^{3/2}} z_3$$

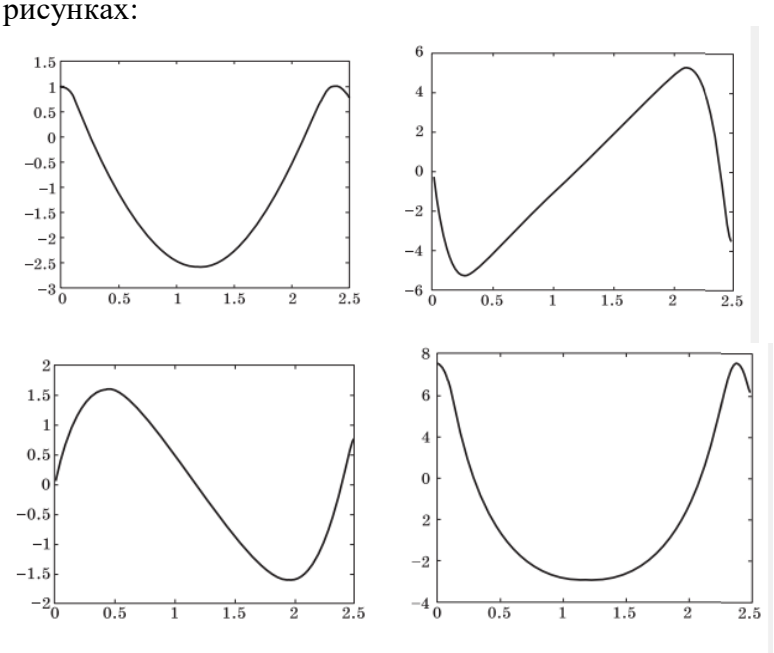
Следуя общим правилам MATLAB решения систем ОДУ первого порядка, создадим m-функцию Orbit.m, возвращающую значения координат вектора-функции, стоящей в левой части системы

```
function dy=Orbit(t,z)
  dy=zeros(4,1); % задание вектора-столбца
                 % размерности 4x1
  % задание координат вектора-функции,
  % стоящей в правой части (4.30)-(4.33)
  dy(1)=z(2);
  dy(2)=-4*pi^2*z(1)/(z(1)^2+z(3)^2)^(3/2);
  dy(3)=z(4);
  dy(4)=-4*pi^2*z(3)/(z(1)^2+z(3)^2)^(3/2);
```

Далее необходимо выполнить в командном окне следующую последовательность команд, сохраненную нами в файле Glava4_1.m:

```
% задание начальных условий  x(0), x'(0), y(0), y'(0)
>>  x0=1;
>>  y0=0;
>>  vx0=0;
>>  vy0=2*pi*1.2;
% вычисление численного
% решения системы (4.30)-(4.33)
>>  [t,Y]=ode45('Orbit',[0:10^-3:2.5],[x0 vx0 y0 vy0]);
>>  figure(1); plot(t,Y(:,1)); % построение зависимости x(t)
>>  figure(2); plot(t,Y(:,2)); % построение зависимости x'(t)
>>  figure(3); plot(t,Y(:,3)); % построение зависимости y(t)
>>  figure(4); plot(t,Y(:,4)); % построение зависимости y'(t)
>>  figure(5); plot(Y(:,1),Y(:,3)); % построение траектории
                                    % движения тела
```

Зависимости кинематических характеристик от времени, а также и траектория движения представлены на рисунках:

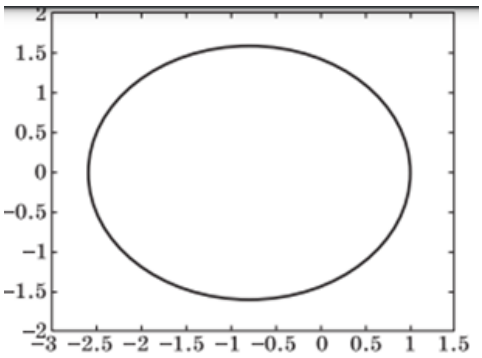


Для отображения на одном чертеже траектории материальной точки и притягивающего центра необходимо выполнить следующую последовательность команд:

```
>>  X0=0;Y0=0; % задание координат притягивающего центра
>>  plot(Y(:,1),Y(:,3),X0,Y0,'o','MarkerSize',5).
```

Для вычисления эксцентриситета орбиты необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava4_2.m

```
>>  MinX=min(Y(:,1)); % вычисление минимального элемента
                      % в первом столбце матрицы Y
>>  MaxX=max(Y(:,1)); % вычисление максимального элемента
                      % в первом столбце матрицы Y
>>  a=(MaxX-MinX)/2; % вычисление длины первой полуоси
```




```
>> MinY=min(Y(:,3)); % вычисление минимального элемента
                     % во втором столбце матрицы Y
>>  MaxY=max(Y(:,1)); % вычисление максимального элемента
                      % во втором столбце матрицы Y
>>  b=(MaxY-MinY)/2; % вычисление длины второй полуоси
% вычисление эксцентриситета
>>  if b<=a
          e=(1-(b/a)^2)^0.5
      else
           e=(1-(a/b)^2)^0.5
      end
e =
    0.4386
```

Типовые задания

Задача 1. Используя описанную программу, проведите исследование влияния возмущений на орбитальную станцию, движущуюся по круговой орбите вокруг Земли: 1. Как меняется орбита станции, если за счет включения на короткое время двигателя пристыкованной к ней ракеты-носителя станция получила приращение скорости в радиальном направлении? 2. Как зависит это изменение от величины приращения скорости в радиальном направлении и его длительности? 3. Установите, меняется ли угловой момент и полная энергия под действием радиального возмущения? 4. Будет ли орбита устойчивой, то есть будет ли малый переданный радиальный импульс вызывать малые возмущения орбиты, а также будет ли орбита оставаться периодической при отсутствии возмущений при дальнейшем движении? 5. Проведите исследования устойчивости в случае движения в тела в потенциале $U(|\vec{r}|) = 1\,|\vec{r}|^{\,2}$под действием радиальных сил. 6. Как меняется орбита станции, если за счет включения на короткое время двигателя пристыкованной к ней ракеты-носителя станция получила приращение скорости в касательном (тангенциальном) направлении? 7. Как зависит это изменение от величины приращения скорости в тангенциальном направлении и его длительности? 8. Установите, меняется ли угловой момент и полная энергия под действием тангенциального возмущения? 9. Будет ли орбита устойчивой, то есть будет ли малый переданный тангенциальный импульс вызывать малые возмущения орбиты, а также будет ли орбита оставаться периодической при отсутствии возмущений при дальнейшем движении? 10. Исследуйте устойчивость орбиты в случае движения в тела в потенциале $U(|\vec{r}|) = 1\,|\vec{r}|^{\,2}$под действием тангенциальных сил.

Задача 2. Известно, что от Солнца в направлении Земли существует поток частиц, называемый солнечным ветром, который приводит к возмущению орбиты движущегося тела. Воздействие солнечного ветра на движущееся тело можно рассматривать как действие слабой постоянной силы W, действующей на тело в горизонтальном направлении. 1. Запишите уравнения движения, описывающие движение тела в гравитационном поле при учете солнечного ветра. 2. Промоделируйте орбиту при

различных значениях величины W. 3. Как меняется орбита тела? 4. Что происходит с интегралами движения при учете солнечного ветра?

ТЕМА 5. МОДЕЛИРОВАНИЕ СТАТИЧЕСКИХ ЭЛЕКТРИЧЕСКИХ И МАГНИТНЫХ ПОЛЕЙ

Электростатическое поле системы неподвижных электрических зарядов

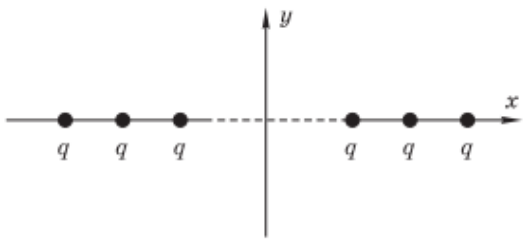
При анализе электростатических полей системы произвольно расположенных зарядов, характеризующихся скалярной функцией – потенциалом $\varphi(\vec{R})$ и векторной функцией — напряженностью $\vec{E}$, возникает задача наглядного представления этих величин. Один из возможных способов представления потенциала электростатического
поля реализуется следующей последовательностью действий:

1) задание функции, возвращающей значения потенциала, вычисляемые в соответствии (5.4) в узлах заданной координатной сетки;
2) задание дискретной координатной сетки;
3) вычисление с учетом силы взаимодействия двух точечных зарядов значений $\varphi(\vec{R})$ в каждом узле координатной
сетки;
4) построение графика поверхности и карты эквипотенциалей.

Следуя последовательности действий, описанной выше, создадим m-файл, содержащий описание функции $\varphi(\vec{R})$:

```
function Phi=Potential(q,xq,yq,X,Y);
% задание функции, возвращающей
% значения потенциала в узлах
% координатной сетки,
% вычисляемых в соответствии с (5.4)
% q - вектор, содержащий значения
% электрических зарядов
% xq, yq - векторы, содержащие x-е и y-е координаты
% электрических зарядов
% X, Y - векторы, содержащие
% x-е и y-е координаты узлов сетки

e0=8.85*10^-12; % диэлектрическая проницаемость вакуума
Nq=length(q); % число зарядов в системе
Nx=length(X); % число узлов по оси oX
Ny=length(Y); % число узлов по оси oY
% проход по каждому узлу сетки
for i=1:Ny
   for j=1:Nx
      s=0;
      % суммирование по зарядам
      for k=1:Nq
         s=s+q(k)/((X(j)-xq(k))^2+(Y(i)-yq(k))^2)^0.5;
      end;
      M(i,j)=s/(4*pi*e0);
   end;
end;
```

Функция Potential.m возвращает матрицу размерности Ny×Nx, содержащую значения потенциала в соответствующих узлах координатной сетки.



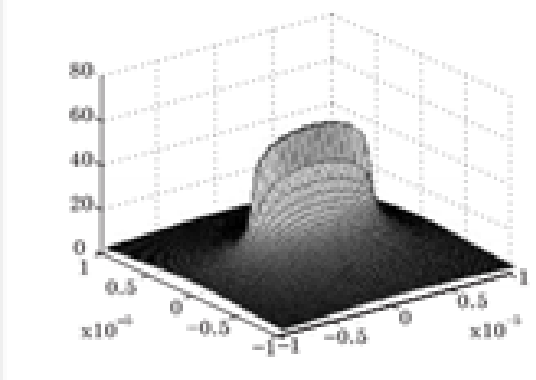
К вычислению потенциала, создаваемого линейной системой точечных зарядов

Вычислим потенциал, создаваемый системой, состоящей из N = 50 электрических зарядов, расположенных в точках с координатами $\left(-5R_0 + \frac{10R_0}{N}i\right), y_i = 0, i = 0, \dots, N$

Для проведения вычислений в качестве единиц измерения заряда будем использовать заряд электрона e = 1, 6 · 10^{-16} Кл, единиц измерения длины – $R_0 = 10^{-6}$ м.

Решение данной задачи в MATLAB выполняется следующей последовательностью команд, сохраненной нами в файле Glava5_1.m:

```
>>  e=1.6*10^-16; % заряд электрона
>>  R0=10^-6; % единица измерения расстояния
>>  N=50; % число зарядов
>>  i=1:N;
>> q(i)=e; % заполнение вектора,
           % содержащего значения зарядов
>> x1=-5*R0; % x-я координата левого
             % конца системы зарядов
>> x2=5*R0; % x-я координата правого конца
            % системы зарядов
>> xq=x1+(x2-x1)/N*i; % вычисление x-х координат
                      % системы зарядов
|>> yq=zeros(1,N); % задание y-х координат
                   % системы зарядов
>> N1=79; % число узлов прямоугольной
          % координатной сетки
>> Xmin=-10*R0; Ymin=-10*R0; % задание координат
                             % нижнего левого
                             % угла координатной сетки
>> Xmax=10*R0; Ymax=10*R0; % задание координат
                           % верхнего правого
                           % угла координатной сетки
>> i=1:N1;
>> X=Xmin+(Xmax-Xmin)/N1*i; % вычисление x-х координат
                            % узлов сетки


>> j=1:N1;
>> Y=Ymin+(Ymax-Ymin)/N1*j; % вычисление y-х координат
                           % узлов сетки
>> M=Potential(q,xq,yq,X,Y); % вычисление значений потенциала
                            % в узлах координатной сетки
>> [X1,Y1]=meshgrid(X,Y); % вычисление матриц X1,Y1,
                         % используемых функциями визуализации
                         % двумерных и трехмерных зависимостей
>> surf(X1,Y1,M); % визуализация поверхности
```



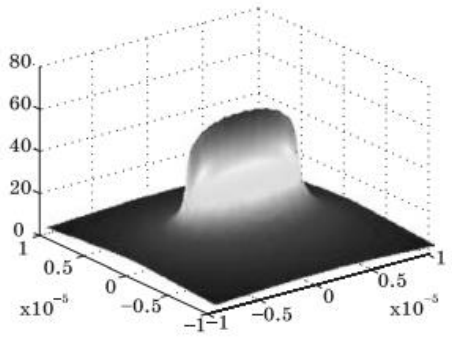
Визуализация потенциала, создаваемого системой линейных зарядов с помощью функции surf

Результат выполнения приведенной выше последовательности команд представлен на рисунке. Для построения отображения поверхности без изображения сетки и заполнения поверхности с использованием интерполяции цвета дополнительно следует ввести команды:

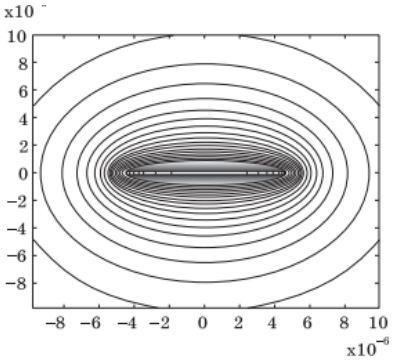
>> figure(2);
>> surf(X1,Y1,M);
>> shading interp

Для построения двумерной карты линий уровня (эквипотенциалей) функции $\varphi = \varphi(x, y)$ следует ввести команды:

```
>> figure(3);
>> contour(X1,Y1,M,33)
% здесь 33 - число
% линий уровня
```

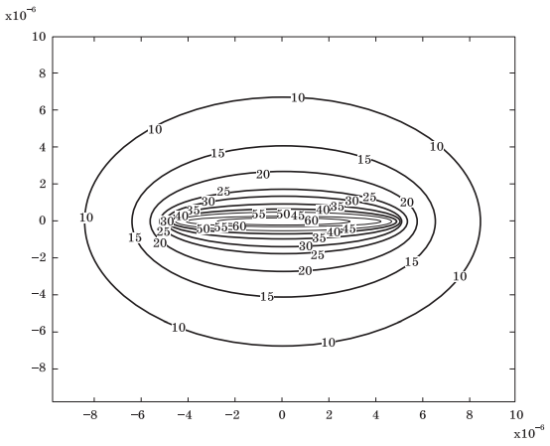


Визуализация потенциала, создаваемого системой линейных зарядов (результат выполнения команды shading interp)



Карта эквипотенциалей функции $\varphi = \varphi(x, y)$

Для вывода значений соответствующей каждой эквипотенциали необходимо выполнить следующую последовательность команд:

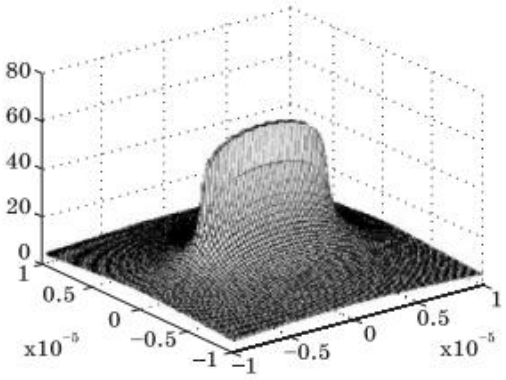


```
>> figure(4);
>> [C,h] = contour(X1,Y1,M)
>> clabel(C,h)
>> colormap cool
```

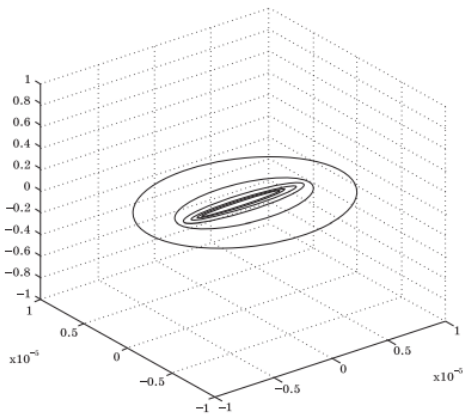
Здесь команда contour возвращает матрицу C, содержащую координаты вершин ломаных, аппроксимирующих соответствующие эквипотенциальные линии, вектор-столбец h дескрипторов для каждой линии уровня, используемый далее командой clabel, и строит карту линий уровня. Далее команда clabel, используя данные, возвращенные функцией contour, подписывает значения функции на каждой линии уровня. Команда colormap определяет цвет заливки карты линий уровня.

Для отображения на одном чертеже поверхности и карты линий уровня следует выполнить команды:

```
>> figure(5);
>> meshc(X1,Y1,M)
```



Результат выполнения функции meshc(X1,Y1,M)

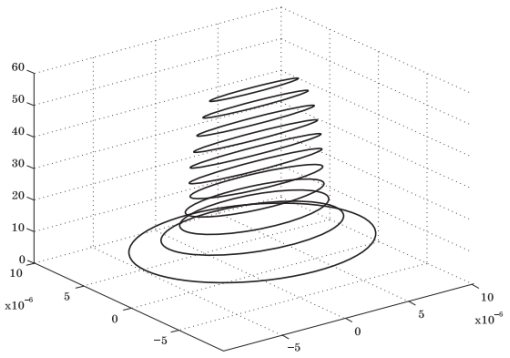


Проекция карты линий уровня на плоскость z = 0

В связи с тем, что поверхность, представленая на рисунке, практически полностью закрывает карту линий уровня, мы перешли в режим редактирования графика (Edit Plot) и, предварительно выделив щелчком мыши изображение поверхности, удалили ее. Результат выполнения описанной выше последовательности действий представлен на рисунке.

Для построения трехмерной карты линий уровня (эвипотенциалей) функции φ = φ(x, y) следует выполнить команды:

```
>> figure(6);
>> contour3(X1,Y1,M,11);
```



Проекция карты линий уровня на плоскость z = 0

В отличие от функции $\vec{\varphi} = \varphi(\vec{R})$, напряженность электростатического поля $\vec{E}(\vec{R})$ является векторной функцией, которая в каждой точке пространства характеризуется величиной поля и направлением. Для описания векторного поля будем использовать силовые линии — линии, касательные к которым в каждой точке параллельны вектору напряженности электрического поля. В связи с тем, что силовые линии несут только информацию о направлении вектора напряженности электрического поля, но не его величине, для анализа изменения величины напряженности электрического поля в пространстве будем использовать функцию $|\vec{E}(\vec{R})|$.

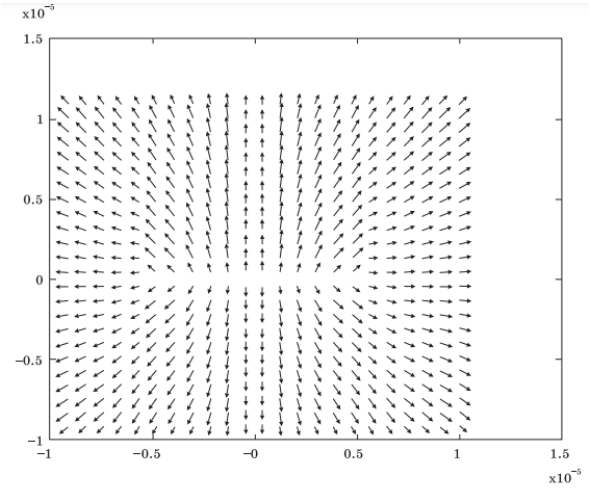
Таким образом, для исследования особенностей напряженности электрического поля, создаваемого произвольной конфигурацией электрических зарядов, следует:

1) задать в пространстве дискретную координатную сетку;

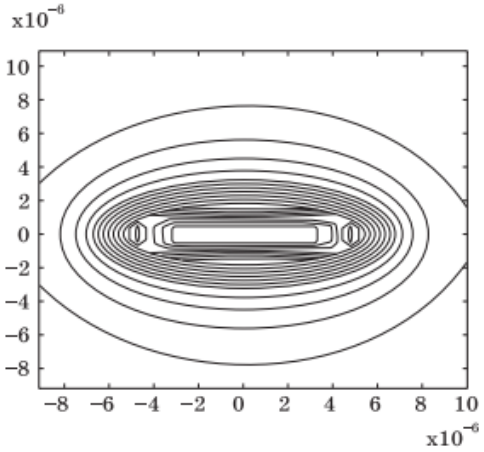
2) вычислить в узлах сетки координаты напряженности электрического поля $\vec{E}(\vec{R}) = (Ex, Ey)$;

3) построить в каждом узле сетки единичные векторы $\vec{n} = (E_x/|E|, E_y/|\vec{E}|)$, касательные к силовой линии электрического поля; 4) построить график поверхности и карту линий уровня функции $|\vec{E}(\vec{R})|$. Данный алгоритм в MATLAB может быть реализован следующей последовательностью команд, сохраненной нами в файле Glava5_2.m:

```
>> e=1.6*10^-16; % заряд электрона
>> R0=10^-6; % единица измерения расстояния
>> N=50; % число зарядов
>> i=1:N;
>> q(i)=e; % заполнение вектора,
           % содержащего значения зарядов
>> x1=-5*R0; % x-я координата левого конца
             % системы зарядов
>> x2=5*R0; % x-я координата правого конца
            % системы зарядов
>> xq=x1+(x2-x1)/N*i; % вычисление x-х координат
                      % системы зарядов
|>> yq=zeros(1,N); % задание y-х координат
                   % системы зарядов
>> N1=23; % число узлов прямоугольной
          % координатной сетки
>> Xmin=-10*R0; Ymin=-10*R0; % задание координат
                             % нижнего левого
                             % угла координатной сетки
>> Xmax=10*R0; Ymax=10*R0; % задание координат
                           % верхнего правого
                           % угла координатной сетки
>> i=1:N1;
>> X=Xmin+(Xmax-Xmin)/N1*i; % вычисление x-х координат
                            % узлов сетки
>> j=1:N1;
>> Y=Ymin+(Ymax-Ymin)/N1*j; % вычисление y-х координат
                            % узлов сетки
>> M=Potential(q,xq,yq,X,Y); % вычисление значений потенциала в
                             % узлах координатной сетки
>> [X1,Y1]=meshgrid(X,Y); % вычисление матриц X1,Y1,
                          % используемых функциями визуализации
                          % двумерных и трехмерных зависимостей
>> [px,py]=gradient(-M,0.1,0.1); % вычисление градиента функции
% вычисление координат единичных векторов, касательных
% к силовой линии
>> px1=px./(px.^2+py.^2).^0.5;
>> py1=py./(px.^2+py.^2).^0.5;

>> figure(1);
>> quiver(X1,Y1,px1,py1,0.5) % построение векторного
                             % поля единичных
                             % векторов, касательных к
                             % силовой линии,
                             % в узлах координатной сетки
>> mp=(px.^2+py.^2).^0.5; % вычисление абсолютных
                          % значений вектора
                          % напряженности
                          % в узлах координатной сетки
>> figure(2);
>> contour(X1,Y1,mp,17); % построение линий
                         % равной напряженности
```



Визуализация силовых линий напряженности электростатического поля, создаваемого линейной системой зарядов



Карта линий равной напряженности электростатического поля, создаваемого линейной системой зарядов

Типовые задания

Задача 1. Создайте документ, позволяющий строить потенциал и напряженность электрического поля диполя, считая, что составляющие его электрические заряды одинаковы по величине и противоположны по знаку.

1. Имеет ли в данной задаче значение, в каких единицах измеряется заряд и расстояние?

2. Как изменится электрическое поле, если заряды будут иметь одинаковые знаки?

3. Убедитесь в том, что силовые линии никогда не заканчиваются на положительных зарядах и всегда идут к отрицательным зарядам.

Почему силовые линии никогда не пересекаются?

Задача 2. Создайте файл, позволяющий строить потенциал и напряженность электрического поля протяженного диполя, то есть диполя, состоящего из N (>10) зарядов, знаки которых последовательно чередуются.

1. Сравните электрическое поле при четном и нечетном числе зарядов, составляющих диполь, в ближней и дальней зонах.

2. Сравните электрическое поле при четном и нечетном числе зарядов, составляющих диполь, с электрическим полем точечного заряда.

3. Исследуйте зависимость электрического поля протяженного диполя в ближней и дальней зонах от расстояния между зарядами.

ТЕМА 6. МОДЕЛИРОВАНИЕ ДВИЖЕНИЯ ЭЛЕКТРИЧЕСКИХ ЗАРЯДОВ В ПОСТОЯННЫХ ЭЛЕКТРИЧЕСКИХ И МАГНИТНЫХ ПОЛЯХ

Рассеивание частиц в центральном поле. Опыт резерфорда

Весьма важным является вопрос о выборе длительности временного интервала, на котором находится численное решение системы ДУ движения для частицы в центральном поле, так как только при его правильном выборе можно точно определить угол рассеивания χ. Совершенно очевидно, что длительность временного интервала зависит от начального положения частицы и начальной скорости, а также величин электрических зарядов взаимодействующих частиц и должна подбираться для конкретных параметров моделируемой системы. Для выбора оптимальной длительности временного интервала можно использовать следующий алгоритм, основанный на постоянстве угла между вектором скорости и осью OX, в областях невозмущенного движения частицы:

1. Найти численное решение системы ДУ движения для частицы в центральном поле на временном интервале длительностью $\approx 2\tilde{x}(0)/\widetilde{v_x}(0)$
2. Построить зависимость угла между вектором скорости и осью OX $\chi = \chi(t)$ от времени:

$$\chi(t) = \arccos(\widetilde{v_x}(t) / \sqrt{(\widetilde{v_x}(t))^2 + (\widetilde{v_y}(t))^2}$$

3. При отсутствии участков, на которых $\chi(t) = const$ ищется численное решение на временном интервале большей длительности и повторно проводится анализ зависимости $\chi = \chi(t)$.

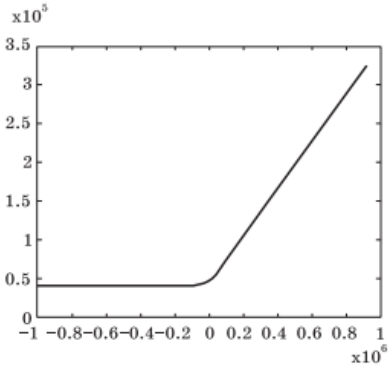
В связи с тем, что в ходе решения рассматриваемой задачи приходится решать систему дифференциальных уравнений, на первом шаге решения следует создать mфайл, содержащий описание функции, возвращающей значения правой части

системы ДУ движения для частицы в центральном поле. Далее приводится листинг соответствующего файла CenterPole.m.

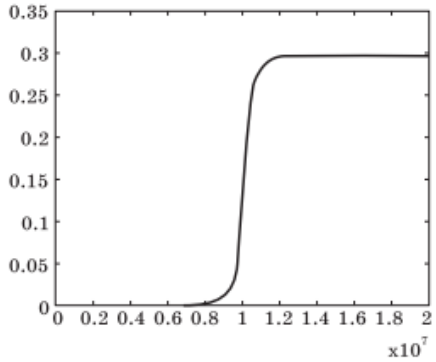
```
% листинг файла CenterPole.m
function dy=CenterPole(t,z)
% описание функции,
% возвращающей значения
% правой части системы (6.17)

global q Q M % задание глобальных переменных
             % для получения
             % их значений
             % из вызывающей программы
A=1.53*q*Q/M;
dy=zeros(4,1); % задание вектора-столбца
               % размерности 4x1
% задание координат вектора-функции,
% стоящей в правой части (6.17)
dy(1)=z(2);
dy(2)=A*z(1)/(z(1)^2+z(3)^2)^(3/2);
dy(3)=z(4);
dy(4)=A*z(3)/(z(1)^2+z(3)^2)^(3/2);
```

Для нахождения численного решения системы ДУ движения для частицы в центральном поле необходимо выполнить приведенную ниже систему команд, сохраненную нами в файле Glava6_1.m:

```
% листинг файла Glava6_1.m
global q M Q % задание глобальных переменных
q=2; % заряд рассеиваемой частицы
M=4; % масса рассеиваемой частицы
Q=79; % заряд рассеивающего центра
x0=-10^6; % x-я координата начальной точки
y0=4*10^4; % значение прицельного параметра
vx0=0.1; % x-я составляющая
         % начальной скорости
vy0=0; % y-я составляющая
       % начальной скорости
Tmax=2*10^7 % длительность временного интервала,
            % на котором ищется численное решение системы ДУ (6.17)
dt=Tmax/2000; % шаг интегрирования системы ДУ (6.17)
% решение системы ДУ
[T Y]=ode45('CenterPole',[0:dt:Tmax],[x0 vx0 y0 vy0]);
% визуализация численного решения ДУ (6.17)
figure(1);
plot(Y(:,1),Y(:,3)); % построение траектории движения

xi=acos(Y(:,2)./(Y(:,2).^2+Y(:,4).^2).^0.5); % вычисление угла
                                             % наклона вектора
                                             % скорости к оси OX
figure(2);
plot(T,xi) % построение зависимости
           % угла рассеивания
           % от времени
```



Траектория движения частицы, рассеиваемой на неподвижном ядре



Зависимость угла рассеивания от времени

Типовые задания

Задача 1. Проведите численные расчеты и постройте на одном чертеже семейство траекторий и углов рассеивания альфа-частицы при различных значениях прицельного параметра. Объясните полученные результаты. Постройте зависимость угла рассеивания от прицельного параметра $\chi=\chi(\rho)$. (Для этого можно выбрать диапазон изменения прицельного параметра, выбрав шаг, разбить его на конечное число точек и для каждого значения прицельного параметра, решив уравнения движения, определить угол рассеивания.) Объясните, почему угол рассеивания является возрастающей или убывающей функцией прицельного параметра.

Задача 2. Для упрощения процесса вычислений и построения графиков зависимостей, характеризующих процесс рассеивания заряженной частицы на неподвижном центре, разработайте проект графического интерфейса пользователя и создайте его практическую реализацию, используя средства визуального проектирования MATLAB.

ТЕМА 7. ФУРЬЕ АНАЛИЗ НЕПРЕРЫВНЫХ И ДИСКРЕТНЫХ ФУНКЦИЙ

Эффект Гиббса

Эффект Гиббса возникает при восстановлении разрывных функций с помощью усеченного ряда Фурье. Изучим основные его проявления на примере разложения прямоугольной импульсной функции.

$$S(t) = \begin{cases} -\frac{1}{2}, & \text{если } 0 \le t \le \frac{T}{2} \\ \frac{1}{2}, & \text{если } \frac{T}{2} \le t \le T \end{cases}$$

которая имеет скачок, равный 1 в точке разрыва T/2.

Формальное разложение в ряд Фурье находится подстановкой функции s(t) в формулы:

$$a_0 = \frac{2}{T}\int_0^t s(t)dt,$$

$$a_n = \frac{2}{T}\int_0^T s(t)\cos(2\pi n/T)dt,$$

$$b_n = \frac{2}{T}\int_0^T s(t)\sin(2\pi n/T)dt.$$

вычислением интегралов, оказывающихся равными

$a_k = 0$

$$b_k = \frac{1}{\pi}\frac{[1+(-1)^{k+1}]}{k} = \begin{cases} \frac{2\pi}{k}, k & -\text{четное} \\ 0, & -\text{нечетное} \end{cases}$$

и подстановкой выражений для коэффициентов в

$$s(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos(n\omega t) + b_n \sin(n\omega t))$$

$$s(t) = \frac{2}{\pi}\left[\sin\left(\frac{2\pi}{T}t\right) + \frac{1}{3}\sin\left(\frac{2\pi}{T}3t\right) + \frac{1}{5}\sin\left(\frac{2\pi}{T}5t\right) + \cdots\right] = \frac{2}{\pi}\sum_{k=0}^{\infty}\frac{1}{2k+1}\sin\left(\frac{2\pi}{T}(2k)+1\right)t$$

Несмотря на то что полученное разложение позволяет познакомиться с основными особенностями эффекта Гиббса, мы предварим дальнейшее рассмотрением этого вопроса описанием последовательности команд, позволяющей получить в MATLAB разложение в ряд Фурье произвольных функций. Для решения поставленной задачи необходимо создать следующие m-файлы: FF.m (содержит описание функции, разлагаемой в ряд Фурье), AF.m (описание функции, возвращающей значение заданного коэффициента разложения в ряд Фурье по косинусам), BF.m (описание функции, возвращающей значение заданного коэффициента разложения в ряд Фурье по синусам.

Далее приводятся листинги данных файлов.

```
% листинг файла FF.m
function z=FF(t,T)
% описание функции,
% раскладываемой в ряд Фурье
N=length(t);
for i=1:N
  if t(i)<0
    z(i)=0;
  end;
  if (t(i)>=0)&(t(i)<=T/2)
    z(i)=1/2;
  end;
  if (t(i)>T/2)&(t(i)<=T)
    z(i)=-1/2;
  end;
  if t(i)>T
    z(i)=0;
  end;
end;

% листинг файла AF.m
function z=AF(k,T)
% описание функции, возвращающей значение
% k-го коэффициента
% разложения в ряд Фурье
% функции FF(t,T) по косинусам

dt=T/1000;
t=0:dt:T;

F=FF(t,T).*cos(2*pi*k/T*t); % вычисление значения
                            % подынтегральной функции

% вычисление интеграла
z=2/T*trapz(t,F);

% листинг файла BF.m
function z=BF(k,T)
% описание функции, возвращающей значение
% k-го коэффициента
% разложения в ряд Фурье
% функции FF(t,T) по синусам
```

```
dt=T/1000;
t=0:dt:T;

F=FF(t,T).*sin(2*pi*k/T*t); % вычисление значения
                            % подынтегральной функции

z=2/T*trapz(t,F); % вычисление интеграла
```

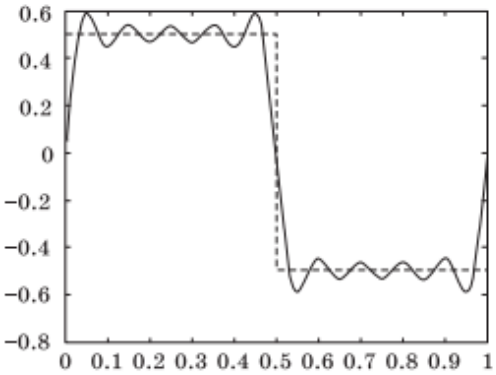
Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava7_1.m:

```
% листинг файла Glava7_1.m
clear all % очистка рабочей области
Nf=9; % число гармоник
k=1:Nf;
T=1; % длительность импульса

A0=AF(0,1); % вычисление A(0)
            % в соответствии с (7.10)

for k=1:Nf  % вычисление коэффициентов
            % A(k), B(k)
  A(k)=AF(k,T);
  B(k)=BF(k,T);
end;

% вычисление значений
% усеченного ряда Фурье
% на временном интервале [0,1]
Np=1000;
t=0:T/Np:1;
for i=1:Np+1
  S=A0/2;
  for k=1:Nf
    S=S+A(k)*cos(2*pi*k/T*t(i))+...
      B(k)*sin(2*pi*k/T*t(i));
  end;
  s(i)=S;
end;
% визуализация усеченного
% ряда Фурье и исходной функции
plot(t,s,t,FF(t,T),'--')
```

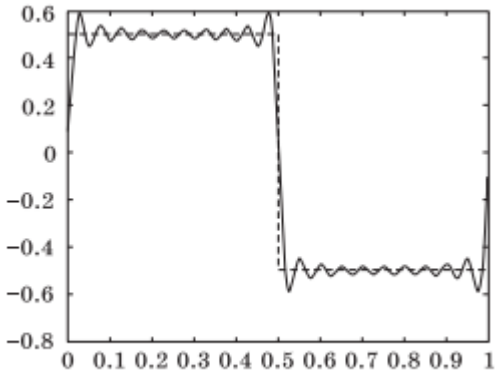
Результаты выполнения описанной выше последовательности команд представлены на рис. 7.2. Сравнение графиков исходной функции и функции, являющейся результатом

разложения в ряд Фурье, представленных на рисунке, обнаруживает их отличие. Данное отличие между исходной функцией и усеченным рядом Фурье носит название эффекта Гиббса .

Для выявления причины обнаруженного эффекта следует изменить в приведенном выше документе число гармоник ряда Фурье и пересчитать документ заново. Графики исходной функции и ее ряда Фурье при использовании 19 гармоник (Nf = 19) представлены на рисунке. Анализ полученных результатов позволяет сделать качественный вывод о том, что увеличение числа членов усеченного ряда Фурье приводит к уменьшению отличия между функциями f(t) и s(t)

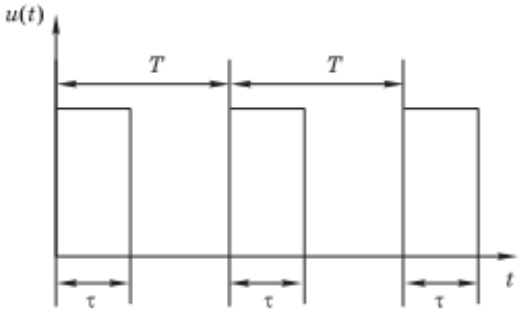


Прямоугольный импульс и частичная сумма ряда Фурье (9 членов ряда)



Прямоугольный импульс и частичная сумма ряда Фурье (19 членов ряда)

Типовые задания



Задача 1. Найдите аналитически коэффициенты ряда Фурье периодической последовательности прямоугольных видеоимпульсов с известными параметрами. 2. В радиотехнике отношение q = T /τ называют скваженностью последовательности. Постройте и сравните зависимости амплитуды спектральных гармоник от частоты nω при различной скваженности последовательности.

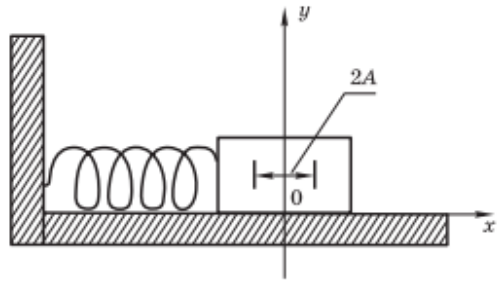
Задача 2 Для оценки количественных характеристик степени отличия между функциями f(t) и s(t) при различном числе членов усеченного ряда Фурье можно использовать максимальное значение функции $f(t) - s(t)$ и дисперсию данной функции. Постройте при различных значениях числа членов усеченного ряда Фурье Nf: 1. Графики функций f(t) − s(t).

2. Оцените, используя построенные ранее графики, периоды колебаний функции f(t)−s(t) и сравните его с периодами колебаний последнего удержанного члена ряда Фурье. 3. Постройте зависимости максимального значения функции f(t)−s(t) и ее среднеквадратичного отклонения от числа членов усеченного ряда Фурье на временном интервале [0, 05; 0, 045]. Примечание: для вычисления среднеквадратического отклонения используйте функцию std(v), v — вектор, содержащий значения функции f(t) − s(t).

ТЕМА 8. МОДЕЛИРОВАНИЕ КОЛЕБАТЕЛЬНЫХ ПРОЦЕССОВ

Линейный гармонический осциллятор

Один из наиболее распространенных типов движения механических систем представляет собой малые колебания, которые система совершает вблизи положения устойчивого равновесия. Наиболее простым с математической точки зрения оказывается описание движения систем, имеющих одну степень свободы. Примером такой системы является тело массой m, расположенное на абсолютно гладкой горизонтальной поверхности и прикрепленное к свободному концу пружины жесткостью k.



К построению модели линейного гармонического осциллятора

Будем описывать положение тела координатой x и примем за начало отсчета (x=0) точку равновесия груза на пружине. При смещении тела из положения равновесия x=0 на небольшое расстояние x, со стороны пружины на тело будет действовать возвращающая сила, пропорциональная величине смещения и направленная в сторону положения равновесия:

$$F = -kx$$

Знак минус в указывает на то, что сила стремится вернуть тело в положение равновесия. На основании второго закона Ньютона

$$m\ddot{x} = -kx$$

Разделив обе части уравнения на массу m и введя обозначение

$$\omega_0 = \sqrt{\frac{k}{m}}$$

запишем уравнения движения линейного гармонического осциллятора в виде

$$\ddot{x} + \omega_0^2 x = 0$$

Где ω_0 называется циклической частотой. Уравнение является линейным дифференциальным уравнением второго порядка. Аналитическое решение данного уравнения хорошо известно:

$$x(t) = A\cos(\omega_0 t + \delta)$$

где A, δ — постоянные, называемые амплитудой и начальной фазой, определяются из начальных условий для координаты и скорости:

$$x_0 = x(0)$$
$$v_0 = v(0)$$

По определению период периодической функции есть наименьшее время, через которое движение повторяется, то есть$x(t+T) = x(t)$

Так как период функции косинус равен 2π, то величина ω0T соответствует одному периоду колебания:

$$\omega_0 T = 2\pi$$

откуда находим связь между ω и T:

$$T = \frac{2\pi}{\omega_0} = 2\pi\sqrt{\frac{m}{k}}$$

Частота колебаний ν представляет собой число периодов в одну секунду.

Необходимо отметить, что период колебаний зависит от отношения k/m, но не зависит от A и δ. Это означает, что период колебаний линейного гармонического осциллятора не зависит от амплитуды колебаний. Легко показать, что полная энергия линейного гармонического осциллятора E

$$E = \frac{1}{2}mv^2 + \frac{1}{2}kx^2$$

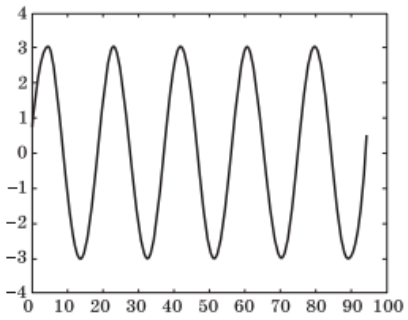
где первое слагаемое — это кинетическая энергия, второе слагаемое — потенциальная энергия. Предваряя рассмотрение более сложных случаев периодического движения физических систем, уравнения движения большинства из которых не допускают

аналитического решения, продемонстрируем общий подход к нахождению и анализу численных решений уравнения движения на примере линейного гармонического осциллятора. Вычисленные зависимости x = x(t), v = v(t) полностью описывают движение одномерного линейного гармонического осциллятора. С другой стороны, пару значений функций {x(tn), y(tn)} в выбранный момент времени можно рассматривать как координаты некоторой точки в двумерном пространстве, называемом фазовой плоскостью. Кривая, образованная на фазовой плоскости, при нанесении точек {x(ti), y(ti)} в последовательные моменты времени называется фазовой траекторий.
Для нахождения численного решения уравнения движения линейного гармонического осциллятора и их визуализации следует привести ДУ второго порядка к системе ДУ первого порядка, затем создать m-файл, содержащий определение функции, стоящей в правой части системы ДУ. Ниже приводится листинг соответствующего файла Oscillator.m

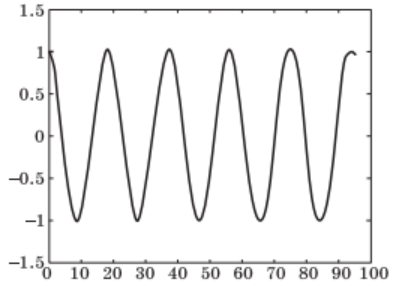
```
% Листинг файла Oscillator.m
function dy=Oscillator(t,z);
% функция, стоящая в правой части
системы ДУ первого порядка,
% эквивалентной ДУ (8.4)
global omega;
dy=zeros(2,1); % задание вектора-столбца
               % размерности 2x1
dy(1)=z(2);
dy(2)=-omega^2*z(1);
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava8_1.m:

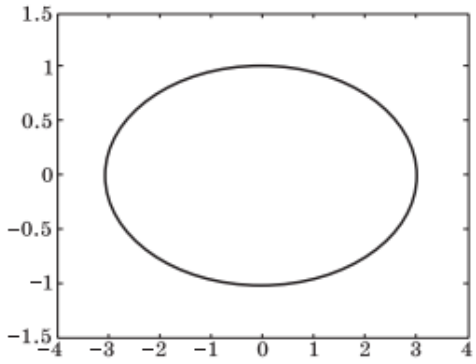
```
% листинг файла Glava8_1.m
global omega;
k=9; % коэффициент жесткости
m=1; % масса осциллятора
T=2*pi*(k/m)^0.5; % период колебаний
omega=2*pi/T; % циклическая частота
R0=[0.5 1]; % начальные условия
N=10^4; % число узлов
        % временной сетки

t Z]=ode45('Oscillator',[0:5*T/N:5*T],R0); % решение системы ДУ

figure(1);
plot(t,Z(:,1)); % визуализация
                % зависимости x=x(t)
figure(2);
plot(t,Z(:,2)); % визуализация
                % зависимости v=v(t)
figure(3); plot(Z(:,1),Z(:,2)); % визуализация фазовой плоскости

figure(4);comet(Z(:,1),Z(:,2)); % динамическая визуализация
                                % движения на фазовой плоскости
```



Зависимость координаты гармонического осциллятора от времени



Зависимость скорости гармонического осциллятора от времени



Фазовая траектория гармонического осциллятора

Типовые задания

Задача 1. Вычислите полную энергию гармонического осциллятора для тех моментов времени, в которые известны значения координаты и скорости. Постройте график временной зависимости величины $\Delta n = (E_n - E_0)/E_0$. Как меняется данная величина во времени? 2. Постройте график зависимости отклонения между численными значениями координаты и скорости и аналитическими. Как меняются данные величины во времени? Оцените точность полученного численного решения по координате и по скорости. 3. Дополните описанный выше документ блоком, позволяющим проводить спектральный анализ численных решений ДУ вычислите спектр зависимостей x = x(t), v = v(t), постройте графики спектральных плотностей $S_x(f)$, $S_y(f)$ и определите, используя данные графики, значения циклической частоты $\omega_0 = 2\pi f_0$ и периода колебаний T. Для различных значений ω0 вычислите, соответствующее им значение периодов T. Предположив, что величина T пропорциональна (k/m) α, оцените показатель степени, построив график зависимости T = T(k/m) в дважды логарифмическом масштабе. 5. Вычислите значения амплитуды A и полной энергии E для начальных условий x(0) = 4, v(0) = 0 и x(0) = 0, v(0) = 4, выбрав в обоих случаях значения k, m, при которых $\omega_0 = 2$. Какая величина определяет значение амплитуды A? 6. Постройте зависимости потенциальной и кинетической энергий от времени. В какие моменты времени эти зависимости достигают своего наибольшего и наименьшего значений? В каких точках находится в данный момент осциллятор? 7. Вычислите и сравните средние значения кинетической и потенциальной энергий за один период колебания. 8. Вычислите зависимости x(t) для различных значений амплитуды A и покажите, что форма данной функции и ее период не зависят от A при фиксированном значении отношения k/m. В каких единицах необходимо измерять время для того, чтобы величина x(t)/A не зависела от A и k/m? Как будет выглядеть уравнение движения линейного гармонического осциллятора (8.4) в данном случае? 9. Будут ли отличаться фазовые траектории для различных начальных условий? Какие физические величины определяют размеры большой и малой полуосей эллипса? Запишите аналитическое уравнение эллипса, проведите его анализ и сравните свои выводы с результатами, полученными при численных расчетах.

ТЕМА 9 МОДЕЛИРОВАНИЕ СИСТЕМ, СОСТОЯЩИХ ИЗ БОЛЬШОГО ЧИСЛА ЧАСТИЦ, МЕТОДОМ БРОУНОВСКОЙ ДИНАМИКИ

Рассматриваемая нами статистическая система является детерминированной, поскольку для описания ее поведения решается задача Коши системы линейных дифференциальных уравнений с постоянными коэффициентами. При этом получаемые решения напрямую зависят от начальных условий (начальной конфигурации системы). Отметим, что их правильный выбор оказывается далеко не простой задачей (например, заранее совершенно не очевидно, как выбрать начальную конфигурацию, чтобы изучаемая система вела себя как жидкость с заданной температурой), поэтому сначала мы обсудим особенности эволюции статистической системы из произвольных начальных конфигураций. Один из возможных вариантов задания начальных условий — размещение частиц в узлах некоторой прямоугольной сетки (размер которой, как очевидно, должен быть меньше размера МД-ячейки) и задание векторов их скоростей случайным образом, например с помощью генератора случайных чисел с равномерным законом распределения. Данный подход использован ниже в задаче моделирования статистической системы методом МД.

Для решения поставленной задачи оказывается удобным сначала создать m-файлы, содержащие описания:

1) функции, возвращающей начальную конфигурацию системы (файл Init.m);

2) функции, возвращающей мгновенные ускорения каждой частицы системы и мгновенное значение потенциальной энергии (файл Acc.m);

3) функции, возвращающей значения координат, составляющих скорости и ускорений вдоль соответствующих координатных осей (файл Verlet.m);

4) функции, возвращающей составной массив, содержащий значения координат, проекций скоростей и ускорений на соответствующие координатные оси в узлах временной сетки (файл Mol_din.m).

```
% листинг функции Init.m
function z=Init(Lx,Ly,Nx,Ny,Vmax)
% функция, возвращающая начальную
% конфигурацию системы
% Lx - ширина МД-ячейки
% Ly - высота МД-ячейки
% Nx - ширина начальной ячейки
% Ny - высота начальной ячейки
% Vmax - максимальное значение скорости

Pos_row=Ly/(Ny+1);
Pos_col=Lx/(Nx+1);
N=Nx*Ny; % число частиц системы
i=1;
for Rows=1:Ny
  for Col=1:Nx
    % задание начальных координат частиц
    x(i)=Pos_col*Col/2;
    y(i)=Pos_row*Rows;
    % задание начальных скоростей частиц
    Vx(i)=Vmax*(2*rand(1)-1);
    Vy(i)=Vmax*(2*rand(1)-1);
    i=i+1;
  end;
end;

% вычисление проекций скорости
% центра масс системы
Vx_full=mean(Vx);
Vy_full=mean(Vy);
i=1:N;
Vx(i)=Vx(i)-Vx_full;
Vy(i)=Vy(i)-Vy_full;

% возврат матрицы, содержащей
% начальные значения координат и
% проекций скоростей частиц
% статистической системы
z=cat(2,x',y',Vx',Vy');

% листинг файла Acc.m
function z=Acc(R_V,Lx,Ly,N)
% функция, возвращающая мгновенные
% ускорения каждой частицы
% системы и мгновенное значение
```

```
% потенциальной энергии
% R_V - матрица, возвращенная функцией Init
% Lx - ширина МД-ячейки
% Ly - высота МД-ячейки
% N - число частиц системы
% инициализация переменных
Ep=0;
for i=1:N
  Ax(i)=0;
  Ay(i)=0;
  Pe(i)=0;
end;

for i=1:N-1
  for j=i+1:N
       Dx=R_V(i,1)-R_V(j,1);
       % проверка граничных условий
       if abs(Dx)>Lx/2
          Dx=Dx-sign(Dx)*Lx;
      end;
      Dy=R_V(i,2)-R_V(j,2);
      % проверка граничных условий
      if abs(Dy)>Ly/2
          Dy=Dy-sign(Dy)*Ly;
       end;
      % вычисление проекции ускорения на ось OX
      Ax(i)=Ax(i)+Fx(Dx,Dy);
      Ax(j)=Ax(j)-Fx(Dx,Dy); % третий закон Ньютона
      % вычисление проекции ускорения на ось OY
      Ay(i)=Ay(i)+Fy(Dx,Dy);
      Ay(j)=Ay(j)-Fy(Dx,Dy); % третий закон Ньютона
      r = (Dx^2+Dy^2).^0.5;
      % вычисление потенциальной энергии
     Ep=Ep+4*(1/r.^12-1/r.^6);
  end;
end;
Pe(N)=Ep;
z=cat(2,Ax',Ay',Pe');

function z1=Fx(x,y)
% проекция силы, действующей на
% частицу вдоль оси OX
z1=-2^(7/3)*(pi^2)/3*x.*(-2/(x.^2+y.^2).^7+...
      1/(x.^2+y.^2).^4);
```

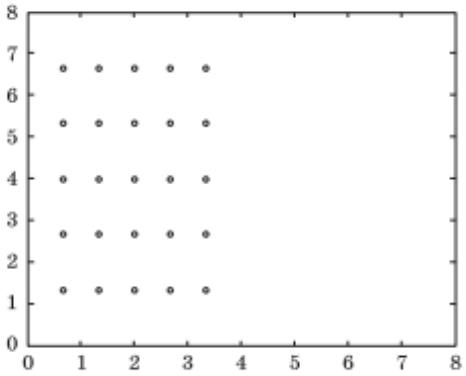
```
function z2=Fy(x,y)
% проекция силы, действующей на
% частицу вдоль оси OY
z2=-2^(7/3)*pi^2/3*y.*(-2/(x.^2+y.^2).^7+...
       1/(x.^2+y.^2).^4);

% листинг файла Verlet.m
function z=Verlet(R_V,Lx,Ly,N,dt)
% функция, возвращающая значения
% координат, составляющих скорости и
% ускорений вдоль соответствующих
% координатных осей
% R_V - матрица, возвращенная функцией Init
% Lx  - ширина МД-ячейки
% Ly - высота МД-ячейки
% N - число частиц системы
% dt - шаг интегрирования

for i=1:N
     % новое положение частицы
     x(i)=R_V(i,1)+R_V(i,3)*dt+R_V(i,5)*(dt^2)/2;
     y(i)=R_V(i,2)+R_V(i,4)*dt+R_V(i,6)*(dt^2)/2;
     % перемещение при необходимости частицы
     % в центральную ячейку
 if x(i)<0
    x(i)=x(i)+Lx;
  end;
  if x(i)>Lx
    x(i)=x(i)-Lx;
  end;
  if y(i)<0
    y(i)=y(i)+Ly;
  end;
  if y(i)>Ly
    y(i)=y(i)-Ly;
  end;
end;
% изменение скорости с использованием
% "старого" ускорения
for i=1:N
  Vx(i)=R_V(i,3)+0.5*R_V(i,5)*dt;
  Vy(i)=R_V(i,4)+0.5*R_V(i,6)*dt;
end;
% вычисление "нового" ускорения
R_Vnew=cat(2,x',y',Vx',Vy');
Accnew=Acc(R_Vnew,Lx,Ly,N);
```

```
% вычисление скорости с использованием
% "нового" ускорения
for i=1:N
     Vx(i)=Vx(i)+0.5*Accnew(i,1)*dt;
    Vy(i)=Vy(i)+0.5*Accnew(i,2)*dt;
end;
z=cat(2,x',y',Vx',Vy',Accnew);

function z1=Fx(x,y)
% проекция силы, действующей на частицу
% вдоль оси OX
z1=-2^(7/3)*(pi^2)/3*x.*(-2/(x.^2+y.^2).^7+1/(x.^2+y.^2).^4);

function z2=Fy(x,y)
% проекция силы, действующей на частицу
% вдоль оси OY
z2=-2^(7/3)*pi^2/3*y.*(-2/(x.^2+y.^2).^7+1/(x.^2+y.^2).^4);

% листинг файла Mol_din.m
function z=Mol_din(Lx,Ly,N,Ni,dt,A)
function z=Mol_din(Lx,Ly,N,Ni,dt,A)
% функция, возвращающая составной массив,
% содержащий значения координат,
% проекций скоростей и ускорений
% на соответствующие координатные оси
% в узлах временной сетки
% Lx - ширина МД-ячейки
% Ly - высота МД-ячейки
% N - число частиц системы
% Ni - число МД-шагов
% dt - шаг интегрирования уравнений движения
% A - массив, возвращенный функцией Init

Accold=Acc(A,Lx,Ly,N);
A2=cat(2,A,Accold);
A3=cat(3,A2);
k=0;
while k<Ni
     A2=Verlet(A2,Lx,Ly,N,dt);
     A3=cat(3,A3,A2);
     k=k+1;
end;
z=A3;
```

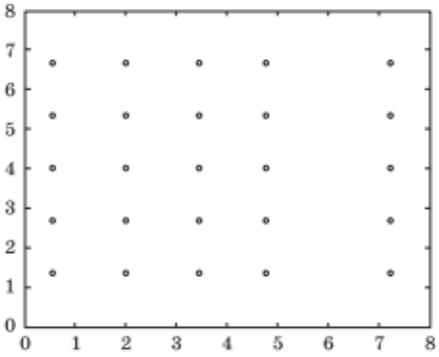
Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava10_1.m:

```
% листинг файла Glava10_1.m
% задание размера начальной ячейки
Nx=5;
Ny=5;
N=Nx*Ny; % число частиц системы
% задание размера МД-ячейки
Lx=8;
Ly=8;

% инициализация начальной конфигурации
Vmax=0.2; % максимальное значение начальной скорости
A=Init(Lx,Ly,Nx,Ny,Vmax);

% задание параметров метода МД
Ni=10^4; % число МД-шагов
dt=1*10^-4; % величина МД-шага

% вызов функции, реализующей метод
% молекулярной динамики
md=Mol_din(Lx,Ly,N,Ni,dt,A);

% визуализация мгновенных конфигураций
% системы в выбранные моменты времени (рис. 10.3-10.6)
figure
plot(md(:,1,1),md(:,2,1),'oK','MarkerSize',3);
axis([0 Lx 0 Ly])

figure
plot(md(:,1,501),md(:,2,501),'oK','MarkerSize',3);
axis([0 Lx 0 Ly])

figure
plot(md(:,1,5001),md(:,2,5001),'oK','MarkerSize',3);
axis([0 Lx 0 Ly])

figure
plot(md(:,1,10001),md(:,2,10001),'oK','MarkerSize',3);
axis([0 Lx 0 Ly])

% вычисление числа частиц
% в левой половине ящика
for i=1:Ni+1
    N1(i)=0;
    tmp=md(:,1,i);
    for j=1:N
```

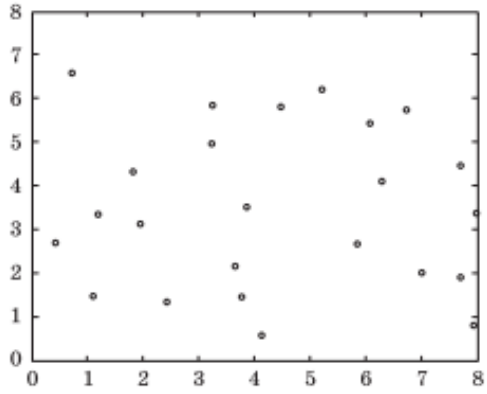
```
        if tmp(j)<=Lx/2
           N1(i)=N1(i)+1;
        end;
    end;
end;
% визуализация зависимости числа частиц,
% находящихся в левой половине
% ящика от времени (рис. 10.7)
i=1:Ni+1;
L(i)=12.5;
figure
plot((i-1)*dt,N1,'k',i*dt,L,'--k');
axis([0 Ni*dt 0 30])

% вычисление мгновенных значений
% кинетической и потенциальной
% энергий системы
for i=1:Ni+1
    Sum=0;
    tmpVx=md(:,3,i);
    tmpVy=md(:,4,i);
    for j=1:N
         Sum=Sum+0.5*(tmpVx(j)^2+tmpVy(j)^2);
    end;
     Ek(i)=Sum;
     Ep(i)=md(N,7,i);
end;

% визуализация зависимостей кинетической,
% потенциальной и полной
% энергий, приходящихся на одну частицу
% от времени (рис. 10.8-10.10)
i=1:Ni+1;

figure
plot((i-1)*dt,Ek/N,'k') % кинетическая энергия

figure
plot((i-1)*dt,Ep/N,'k') % потенциальная энергия

figure
plot((i-1)*dt,(Ek+Ep)/N,'k') % полная энергия
```



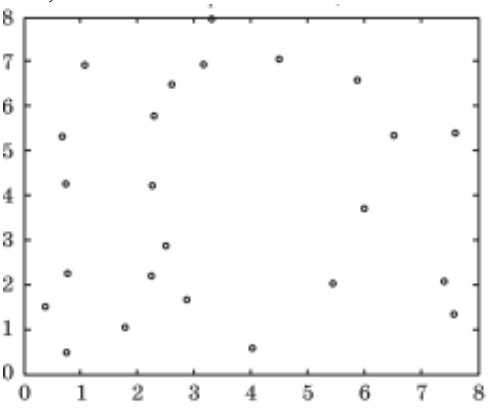
. Начальная конфигурация статистической системы



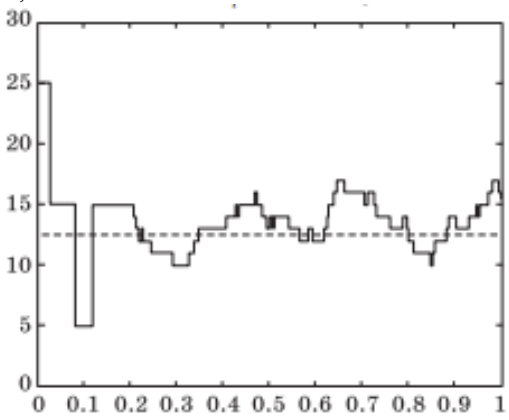
Конфигурация статистической системы в момент времени t = 0, 05



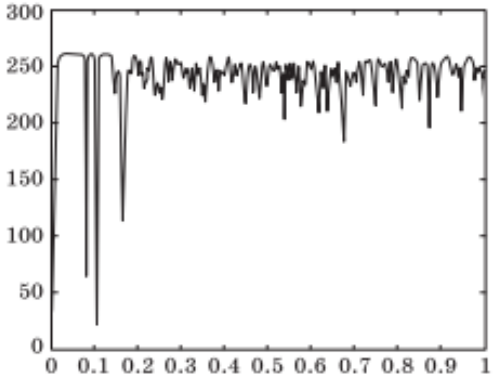
Конфигурация статистической системы в момент времени t = 0, 5



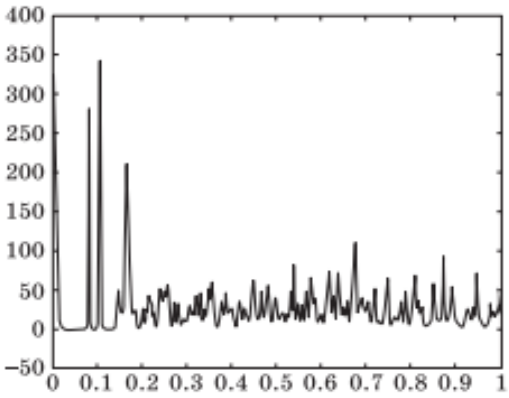
Конфигурация статистической системы в момент времени t = 1, 0

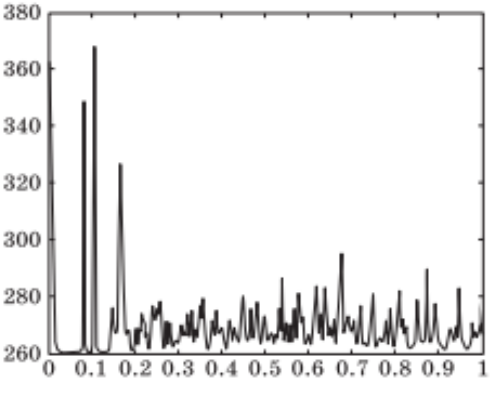


Зависимость числа частиц, находящихся в левой половине ящика, от времени (N lef t = N lef t(t))



Зависимость кинетической энергии, приходящейся на одну частицу, от времени (Ek = Ek(t)





Зависимость потенциальной энергии, приходящейся на одну частицу, от времени (Ep = Ep(t))

Зависимость полной энергии, приходящейся на одну частицу, от времени (E = E(t))

Анализ зависимостей, представленных на рисунках, показывает, что с течением времени система стремится к состоянию равновесия (процесс релаксации), в котором примерно одинаковыми остаются число частиц в левой и правой половинах МД-ячейки и полная энергия системы. (Отметим, что поскольку значения начальной скорости задаются с помощью генератора случайных чисел, данные зависимости, полученные при пересчете документа, будут всегда отличаться от зависимостей, представленных на рисунках, однако качественно они должны вести себя подобным образом.)

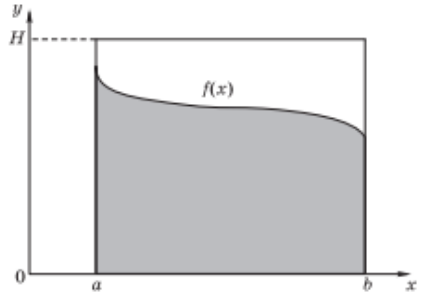
Типовые задания

Задача 1. Используя зависимости Nlef t = Nlef t(t), Ek = Ek(t), Ep = Ep(t), E = E(t), оцените: 1) время релаксации статистической системы к состоянию равновесия; 2) средние значения кинетической, потенциальной и полной энергий системы в состоянии равновесия; 3) оцените величину флуктуаций кинетической, потенциальной и полной энергий системы в состоянии равновесия.

Задача 2. Рассмотрите обратный во времени процесс, выбрав в качестве начальных значений координат частиц, соответствующие координаты из последней вычисленной конфигурации статистической системы Ni(X_i (0) = x_i (dt · Ni), Y_i (0) = yi (dt · N i), i = 0, 1, . . . , N), в качестве начальных скоростей соответствующие значения скоростей из последней конфигурации статистической системы, взятых с противоположным знаком (V x_i(0) = −vx_i(dt · N i), V y_i(0) = −vy_i(dt · N i)). Каково состояние статистической системы в момент времени dt · N i?

ТЕМА 10. МЕТОДЫ МОНТЕ-КАРЛО

Проиллюстрируем идеи метода Монте-Карло на примере вычисления определенного интеграла от функции, зависящей от одной переменной. Пусть нам необходимо вычислить интеграл $F = \int_a^b f(x)dx$от некоторой заданной функции f(x) на интервале [a, b]. В предыдущем разделе мы рассмотрели несколько различных формул интегрирования, в которых использовались значения функции f(x), вычисляемые в равноотстоящих точках. Однако можно использовать и другой подход, суть которого легко понять из следующего примера.



Представим себе прямоугольник высотой H и длиной b − a такой, что функция f(x) целиком лежит внутри данного прямоугольника (рис. 11.4). Сгенерируем N пар случайных чисел равномерно распределенных в данном прямоугольнике:

$$a \le x_i \le b; 0 \le y_i \le H$$

Тогда доля точек *(x_i, y_i),* удовлетворяющих условию $y_i \leq f(x_i)$, является оценкой отношения интеграла от функции f(x) к площади рассматриваемого прямоугольника. Следовательно, оценка интеграла в данном методе может быть получена по формуле

$$F_N = A\frac{n_s}{N}$$

где n_s — число точек, удовлетворяющих условию $y_i \le (x_i)$, N — полное количество точек, A — площадь прямоугольника. В MATLAB описанный алгоритм реализуется следующей последовательностью команд, сохраненную нами в файле Glava11_1.m:

```
% листинг файла Glava11_1.m

% координаты нижнего левого угла прямоугольника
Xmin=0;
Ymin=0;
% координаты правого верхнего угла прямоугольника
Xmax=pi/2;
Ymax=1.5;
N=1000; % число точек
        % случайной последовательности

% вычисление последовательности
% случайных чисел, равномерно
% распределенных на интервале [Xmin,Xmax]
for i=1:N
    x(i)=Xmin+(Xmax-Xmin)*rand(1);
end;

% вычисление последовательности
% случайных чисел, равномерно
% распределенных на интервале [Ymin,Ymax]


 for i=1:N
    y(i)=Ymin+(Ymax-Ymin)*rand(1);
 end;
 f=inline('sin(x)', 'x'); % задание подынтегральной функции

 % подсчет точек, лежащих
 % ниже графика функции f(x)
 >> for i=1:N
          if f(x(i))>=y(i)
             s(i)=1;
          else
             s(i)=0;
          end;
      end;
 >>S=sum(s);
 >> A=(Xmax-Xmin)*(Ymax-Ymin); % площадь прямоугольника
 >> A*S/N %  % вычисление значения интеграла
```

В приведенной выше последовательности команд мы использовали функцию inline, которая преобразует первый аргумент функции, являющейся строковым выражением, в описание функции, зависящей от переменных, стоящих на втором и последующих местах в списке формальных параметров функции. Например, для задания функции.

F(x,y) = sin(x+y)

необходимо выполнить следующую команду

```
>> F=inline('sin(x+y) ', 'x', 'y')
F=
     Inline function:
     f(x,y) = sin(x+y)
```

Типовые задания

Задача 1. Дополните файл, листинг которого приведен выше, блоком команд, позволяющим отображать на одном рисунке график функции и точки, распределенные случайным образом.

Задача 2. Сравните значение определенного интеграла, полученное с помощью описанного выше алгоритма, с его точным значением и оцените погрешность вычислений. 2. Не менее 10 раз пересчитайте документ, фиксируя при этом получаемые значения определенного интеграла. Оцените среднее значение полученной последовательности значений и ее дисперсию. Сравните полученные результаты с точным значением определенного интеграла. 3. Выполните задание 2 для различного числа точек случайной последовательности N. Постройте график зависимости смещения значения определенного интеграла (смещение — разность между точным и вычисленным значениями определенного интеграла) от N. Постройте график зависимости дисперсии от N. Используя метод наименьших квадратов, попробуйте подобрать функцию, описывающую зависимость дисперсии от числа точек N.

ТЕМА 11. СЛУЧАЙНЫЕ БЛУЖДАНИЯ

Метод случайных блужданий на плоскости

Рассмотрим обобщения метода одномерных случайных блужданий для систем большей размерности. Их обсуждение совершенно естественно начать с рассмотрения случайных блужданий в двумерной плоскости. Статический характер задачи случайного блуждания означает, что следует рассматривать либо большое количество случайных блужданий одного пешехода, либо большое число пешеходов, движущихся одновременно. Далее нами используется второй подход.

Рассмотрим группу пешеходов, расположенных в момент времени t = 0, случайным образом в круге единичного радиуса с центром в начале координат. Будем считать, что на каждом шаге по времени каждый пешеход движется случайным образом равновероятно в одном из четырех направлений: в положительном направлении оси OX, в отрицательном направлении оси OX, в положительном направлении оси OY, в отрицательном направлении оси OY. При этом направление движения выбирается в соответствии с методом Монте-Карло в зависимости от величины случайного числа и заданного порогового значения.

Для моделирования двумерных случайных блужданий необходимо создать два файла: файл Init2.m, содержащий описание функции, возвращающей начальную конфигурацию системы, и файл Move.m, содержащий описание функции, возвращающей мгновенные значения координат пешеходов.

```
% листинг файла Init2.m
function z = Init2(NParticle,R)
% функция, возвращающая начальную
% конфигурацию системы
% NParticle - число пешеходов
% R - радиус-вектор максимального смещения
% от начала координат

for i=1:NParticle
   R0=(2*rand(1)-1)*R;
   teta=2*pi*rand(1);
   XY(i,1)=R0*cos(teta);
   XY(i,2)=R0*sin(teta);
end;
z=XY;

% листинг файла Move.m
function z = Move(A,NParticle,NStep)
% функция, возвращающая
% мгновенные значения координат пешеходов
% A - массив, возвращенный функцией Init
% NParticle - число пешеходов
% NStep - число шагов Монте-Карло

for j=1:NParticle
  XY(1,1)=A(j,1);
  XY(1,2)=A(j,2);
  for i=1:NStep-1
    prob=rand(1);
    if prob<=0.25
      XY(i+1,1)=XY(i,1)+1;
      XY(i+1,2)=XY(i,1);
    end;
```

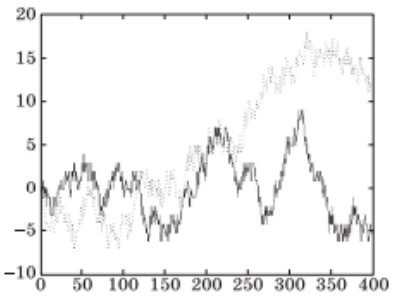
```
    if (prob>0.25)&(prob<=0.5)
      XY(i+1,1)=XY(i,1)-1;
      XY(i+1,2)=XY(i,1);
    end;
    if (prob>0.5)&(prob<=0.75)
      XY(i+1,1)=XY(i,1);
      XY(i+1,2)=XY(i,1)-1;
    end;
    if prob>0.75
      XY(i+1,1)=XY(i,1);
      XY(i+1,2)=XY(i,1)+1;
    end;
  end;
  if j==1
    Z=XY;
  else
    Z=cat(3,Z,XY);
  end;
end;
z=Z;
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava12_4.m:

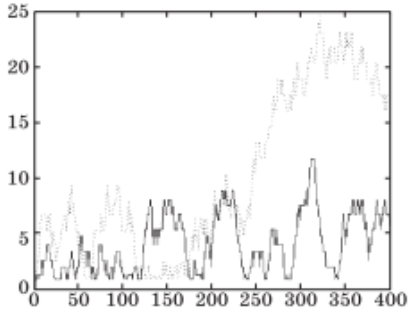
```
% листинг файла Glava12_4.m
NParticle=2000; % число пешеходов
NStep=400; % число шагов Монте-Карло
R=1; % радиус-вектор
     % максимального смещения
     % от начала координат
A=Init2(NParticle,R); % вычисление
                      % начальной конфигурации

% визуализация начальной
% конфигурации системы
Np=2000;
i=1:Np;
dphi=2*pi/(Np-1);phi(i)=dphi*(i-1);
x=cos(phi);y=sin(phi);

figure;
plot(x,y,'k');
hold on
plot(A(:,1),A(:,2),'ok','Markersize',0.1);
hold off
```

```
% вычисление координат каждого
% пешехода в заданные моменты времени
Am=Move(A,NParticle,NStep);

% визуализация мгновенных
% значений координат
% и радиусов-векторов
% первого и сотого пешеходов
i=1:NStep;
figure;
plot(i,Am(:,1,1),'k',i,Am(:,1,100),':k');

figure;
plot(i,Am(:,2,1),'k',i,Am(:,2,100),':k');

R=(Am(:,1,1).^2+Am(:,2,1).^2).^0.5;
R100=(Am(:,1,100).^2+Am(:,2,100).^2).^0.5;
figure;
plot(i,R,'k',i,R100,':k');
% вычисление усредненных по ансамблю
% пешеходов значений координат
for i=1:NStep
    for j=1:NParticle
        X(j)=Am(i,1,j);
        Y(j)=Am(i,2,j);
    end;
    x1(i)=mean(X);
    y1(i)=mean(Y);
end;

% визуализация мгновенных
% значений координат
% и радиуса-вектора,
% усредненных по ансамблю пешеходов
i=1:NStep;
figure
plot(i,x1,'k')

figure;
plot(i,y1,'k')
i=1:NStep;
Rm=(x1.^2+y1.^2).^0.5;
figure;
plot(i,Rm,'k');
```

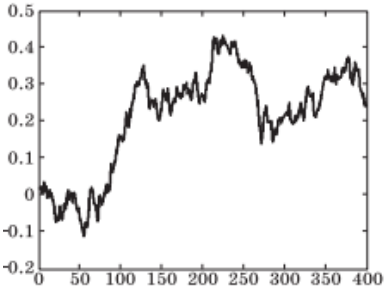
```
% вычисление и визуализация квадратов
% координат и квадрата радиуса-вектора,
% усредненных по ансамблю пешеходов
for i=1:NStep
    for j=1:NParticle
        X(j)=Am(i,1,j).^2;
        Y(j)=Am(i,2,j).^2;
    end;
    xm(i)=mean(X);
    ym(i)=mean(Y);
end;

% вычисление и визуализация мгновенных
% значений среднеквадратичного
% отклонения усредненной траектории
i=1:NStep;
figure
plot(i,xm,'k')

figure;
plot(i,ym,'k')

R2=xm+ym-x1.^2-y1.^2;
figure
plot(i,R2,'k')
```



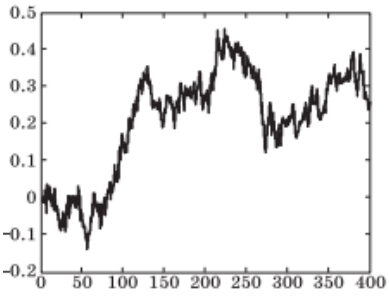
Зависимость мгновенных значений у-й координаты первого и сотого пешеходов от номера шага Монте-Карло



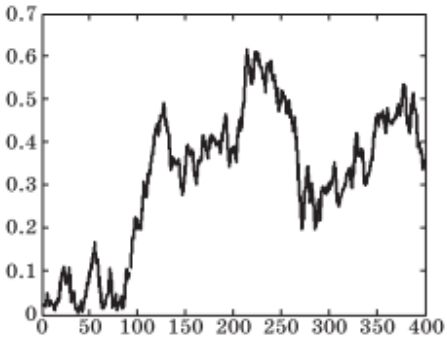
Зависимость мгновенных значений радиусов-векторов первой и сотой частиц от номера шага Монте-Карло



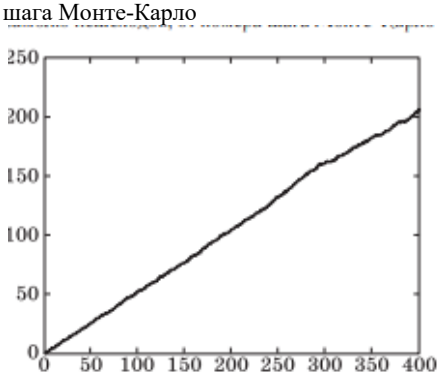
Зависимость мгновенных значений x-й координаты, полученной усреднением по ансамблю пешеходов, от номера шага Монте-Карло



Зависимость мгновенных значений у-й координаты, полученной усреднением по ансамблю пешеходов, от номера шага Монте-Карло

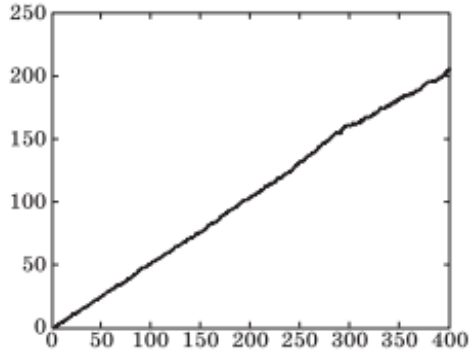


Зависимость мгновенных значений радиуса-вектора, полученного усреднением по ансамблю пешеходов, от



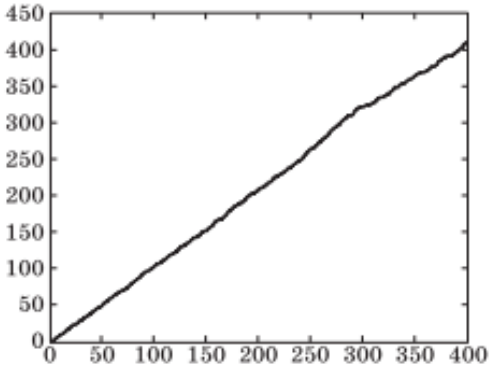
Зависимость мгновенных значений усредненного по ансамблю пешеходов квадрата смещения вдоль оси OX от

номера шага Монте-Карло



Зависимость мгновенных значений усредненного по ансамблю пешеходов квадрата смещения вдоль оси OY от номера шага МонтеКарло

номера шага МонтеКарло



Зависимость мгновенных значений среднеквадратичного отклонения усредненной по ансамблю пешеходов траектории от номера шага Монте-Карло

Анализ зависимости, представленной на рисунках, показывает, что с течением времени происходит линейное увеличение площади окружности, внутри которой находятся все пешеходы, то есть

$$\langle \Delta R_i^2 \rangle = Ai^v$$

где A ~ 1 — коэффициент пропорциональности

видно, что среднеквадратичное смещение $R_i = \sqrt{\langle \Delta R_i^2 \rangle}$ меняется с увеличением числа шагов по закону $\sqrt{\langle \Delta R_i^2 \rangle} = \sqrt{A} i^v$

где для рассматриваемой двумерной модели ν=1/2.

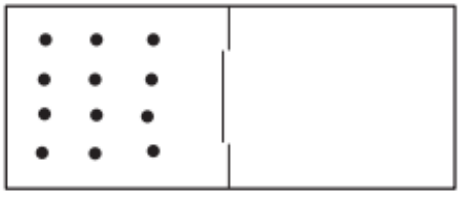
Типовые задания

Задача 1. Создайте анимационный клип, демонстрирующий в динамике процесс увеличения площади, занимаемой пешеходами.

Задача 2 Оцените, используя метод наименьших квадратов, значение коэффициента A в $\langle \Delta R_i^2 \rangle = Ai^v$. Определите погрешность найденного значения A, используя метод оценки погрешностей коэффициентов линейной регрессии.

ТЕМА 12. МОДЕЛИРОВАНИЕ СТАТИСТИЧЕСКОЙ СИСТЕМЫ В ПРОЦЕССЕ РЕЛАКСАЦИИ И СОСТОЯНИИ РАВНОВЕСИЯ

Моделирование процесса релаксации статистической системы



Рассмотрим сосуд, в котором находится идеальный газ, разделенный перегородкой на две равные части, одна из которых в начальный момент времени пуста. В левой половине сосуда находится N молекул. После мгновенного удаления подвижной заслонки молекулы начинают переходить из заполненной части сосуда в пустую.

Поскольку движение каждой частицы происходит независимо от всех остальных, вероятность прохождения через отверстие для всех частиц одинакова. Будем полагать, что в единицу времени через отверстие проходит одна частица.

Для идентификации микро- и макросостояний рассматриваемой системы предположим, что частицы изначально пронумерованы от 1 до N. Микросостояние будем описывать номерами частиц, находящимися в левой половине ящика, поскольку знание номеров этих частиц позволяет однозначно определить номера частиц, находящихся в правой половине ящика. В качестве макроскопической характеристики рассматриваемой системы будем использовать число молекул, находящихся в левой половине ящика. Основная цель исследования состоит в определении численности частиц в левой n и правой n ′ половинах ящика (n+n ′ = N) в заданные моменты времени. Очевидно, что для получения однозначного ответа на данный вопрос требуется знание функции PN (n, t), определяющей вероятность нахождения в момент времени t в левой половине ящика n частиц.

Один из методов нахождения функции PN (n, t) — метод точного перебора. Рассмотрим метод перебора для случая, когда при t=0 n=6 и n′=0. При t=1 возможна единственная комбинация: n=5, n′=1, откуда следует, что P(n=5, t=1) = 1

При t=2 возможны комбинации, когда одна из пяти частиц, оставшихся в левой половине ящика, смещается направо или же частица, находящаяся в правой половине ящика, возвращается в левую половину. Так как первая комбинация может реализоваться пятью различными способами, вероятность данного события P(n=4, t=2)=5/6. Вторая комбинация реализуется единственным способом, поэтому вероятность данного события P(n=6, t=2)=1/6. Следовательно, при t=2 среднее число частиц в левой половине ящика ⟨n⟩ составляет

$$\langle n\rangle = 4P(n=4,t=2) + 6P(n=6,t=2) = 4\frac{2}{6}$$

На следующем шаге по времени имеем

$$P(n=3,t=3) = \frac{4}{6}P(n=4,t=2) = \frac{20}{36}$$

что отвечает перемещению одной из четырех оставшихся частиц вправо. Рассуждая аналогично, найдем

$$P(n=5,t=3) = (1)P(n=6,t=2) + \frac{2}{6}P(n=4,t=2) = \frac{16}{36}$$

Следовательно, при t=3 среднее число частиц в правой половине ящика составляет

$$\langle n\rangle = 5P(n=5,t=3) + 3P(n=3,t=3) = 3\frac{8}{9}$$

Так как полное число частиц мало, перебор удается продолжить еще несколько временных шагов. Однако, как легко видеть, количество комбинаций быстро увеличивается с ростом t. При достаточно больших N и t число возможных комбинаций становится столь велико, что не хватит мощности даже современных компьютеров.

В этих условиях, как и в задаче о случайных блужданиях, весьма эффективным оказывается метод Монте-Карло. Напомним, что в методе Монте-Карло генерируется репрезентативная выборка случайных перемещений, являющаяся подмножеством множества всех возможных перемещений. При этом чем больше окажется объем выборки, тем более высокая точность будет у получаемых результатов.

Для реализации метода Монте-Карло необходимо задать вероятность перехода частицы из левой половины ящика в правую. Так как любая частица с равной вероятностью может пройти через отверстие, вероятность перемещения в единицу времени частицы из левой половины ящика в правую пропорциональна числу частиц в левой половине в данный момент времени, деленному на полное число частиц n/N. Используя данный факт, можно моделировать временную эволюцию модели в

соответствии со следующим алгоритмом, который реализуется следующей последовательностью действий:

1. Задание числа шагов T.
2. Инициализация счетчика *j=0.*
3. Генерация случайного числа Prob в интервале от 0 до 1.
4. Сравнение Prob с текущим значением n/N в левой половине сосуда.
5. Перемещение частицы слева направо, если Prob≤n/N, и перемещение частицы справа налево при Prob> n/N.
6. Повторение п. 3–5, пока j ≥ T.

Для реализации описанного алгоритма в MATLAB необходимо создать два файла:

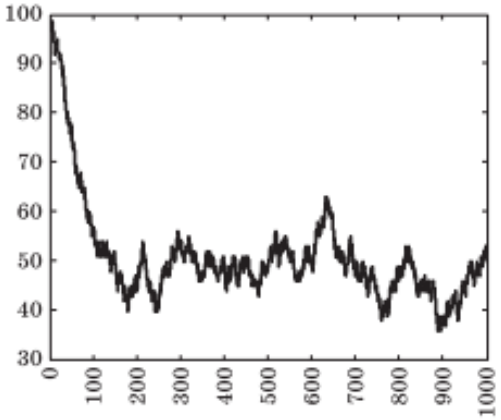
1) файл ParticleAtTheLeft.m, содержащий описание функции, возвращающей единичную реализацию зависимости числа частиц в левой половине ящика n = n(t);

2) Файл Ensemble.m, содержащий описание функции, возвращающей ансамбль реализаций зависимостей n = n(t).

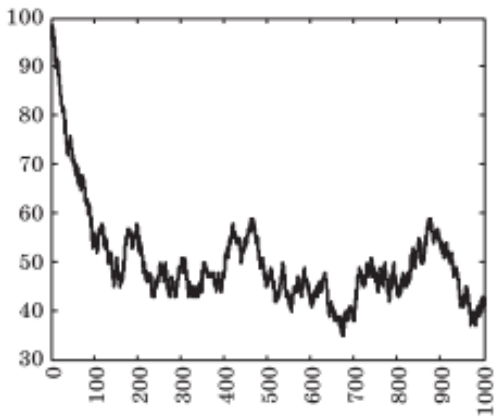
```
% листинг файла ParticleAtTheLeft.m
function z=ParticleAtTheLeft(NParticle,NStep)
% функция, возвращающая
% единичную реализацию зависимости

 % NParticle - число частиц
             % в левой половине ящика
             % в момент времени t=0
 % NStep - длина единичной реализации

 LeftN(1)=NParticle;
 for i=2:NStep
   Prob=LeftN(i-1)/NParticle;
   if rand(1)<=Prob
     LeftN(i)=LeftN(i-1)-1;
   else
     LeftN(i)=LeftN(i-1)+1;
   end;
 end;
 z=LeftN;

 % листинг файла Ensemble.m
 function z = Ensemble(NTrial,NParticle,NStep)
 % NTrial - объем ансамбля реализаций
 % NParticle - число частиц
             % в левой половине ящика
             % в момент времени t=0
 % NStep - длина единичной реализации
 for i=1:NTrial
   Z=ParticleAtTheLeft(NParticle,NStep);
   if i==1
     z=Z;
   else
     z=cat(3,z,Z);
   end;
 end;
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava13_1.m:

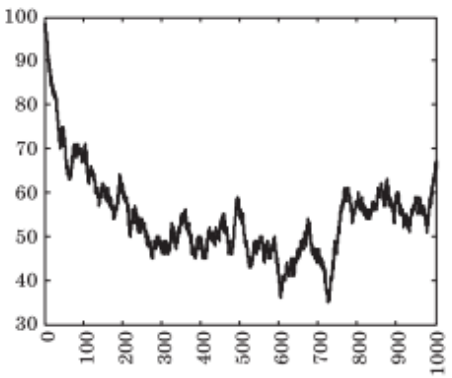
```
% листинг файла Glava13_1.m
NParticle=100; % число частиц системы
NStep=1000; % длина единичной реализации
NTrial=100; % объем ансамбля реализаций

% вычисление
% ансамбля реализаций
E=Ensemble(100,NParticle,NStep);

% визуализация выбранных из ансамбля
% реализаций зависимостей

  i=1:NStep;
  figure(1);
  plot(i,E(1,:,1),'k')

  figure(2);
  plot(i,E(1,:,50),'k')

  figure(3);
  plot(i,E(1,:,100),'k')

  % вычисление средней
  % по ансамблю
  % реализаций зависимости
  for i=1:NStep
      for j=1:NTrial
          s(j)=E(1,i,j);
      end;
      Em(i)=mean(s);
   end;

  % визуализация средней
  % по ансамблю
  % реализаций зависимости
  i=1:NStep;
  figure(4);
  plot(i,Em);
```



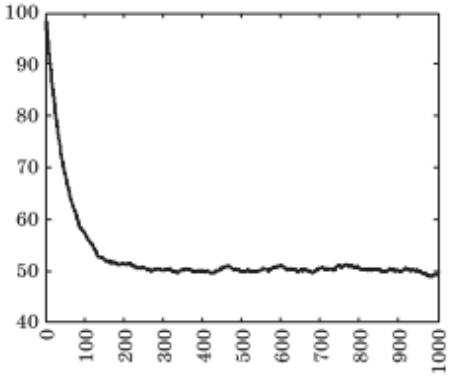
Зависимость числа частиц в левой половине ящика от времени. (Ансамбль № 1)



Зависимость числа частиц в левой половине ящика от времени. (Ансамбль № 50)



Зависимость числа частиц в левой половине ящика от времени. (Ансамбль № 100)

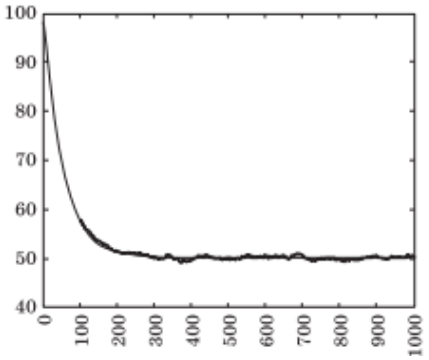


Усредненная по ансамблю реализаций зависимость числа частиц в левой половине ящика от времени

Анализ зависимости n=n(t), представленной на рисунке, показывает, что с течением времени рассматриваемая система стремится к состоянию, в котором в обеих половинах ящика находится примерно одинаковое число частиц. Если пренебречь малыми случайными флуктуациями, то с макроскопической точки зрения (когда нас интересуют не номера частиц, находящихся в левой половине системы, но их количество) можно считать, что начиная с некоторого момента времени параметры системы перестают зависеть от времени. Данное состояние системы называется равновесным. Время, необходимое системе для достижения равновесного состояния, называют временем релаксации $\tau_{rel.}$.

Для оценки τ_{rel} аппроксимируем зависимость n = n(t) аналитической функцией вида $ae^{bt}+c$, дополнив описываемый файл, позволяющий оценивать неизвестные коэффициенты функции, используя метод наименьших квадратов. Для аппроксимации нахождения коэффициентов аппроксимирующей функции в MATLAB необходимо выполнить следующую последовательность команд:

```
% листинг файла Glava13_2.m

NParticle=100; % число частиц системы
NStep=1000; % длина единичной реализации
NTrial=100; % объем ансамбля реализаций

% вычисление ансамбля реализаций
E=Ensemble(NTrial,NParticle,NStep);

% вычисление средней
% по ансамблю реализаций зависимости
for i=1:NStep
      for j=1:NTrial
```



Зависимость n = n(t) и аппроксимирующая ее функция

```
        s(j)=E(1,i,j);
      end;
      Em(i)=mean(s);
 end;
% нахождение функции, аппроксимирующей зависимость
F10=inline('exp(u(1)+u(2).*z)+u(3)','u','z') % задание
                                              % аппроксимирующе
                                              % функции
i=1:NStep;
x(i)=i;
beta=nlinfit(x,Em,F10,[5 -0.1 1]) % функция, возвращающая реше
                                  % системы уравнений метода
                                  % наименьших квадратов
                                  % [5 -0.1 1] - начальные зна
                                  % искомых коэффициентов beta

% визуализация зависимости,
% усредненной по ансамблю
% реализаций, и аппроксимирующей ее функции
plot(x,Em,'k',x,F10(beta,x),'k')
```

Время релаксации изучаемой системы к состоянию равновесия может быть определено из очевидного равенства

$$N_{left} \coloneqq ae^{b\tau_{rel}} + c$$

откуда

$$\tau_{rel} \coloneqq \frac{1}{b}\ln\left(\frac{|\Delta N|}{a}\right)$$

где

$$\Delta N \coloneqq \left|\frac{N_{particle}}{2} - beta(3)\right|$$

Таким образом, для вычисления оценки времени релаксации необходимо дополнить описываемый документ следующими командами:

```
dN=abs(NParticle/2-beta(3));
tau=floor(1/beta(2)*log(dN/beta(1)))
```

Выполнив приведенную выше последовательность команд, получаем

$t_{au}=187$

Типовые задания

Задача 1. Дополните, описанную выше последовательность команд, блоком команд, позволяющим для заданного набора числа частиц рассматриваемой системы проводить автоматические вычисления времени релаксации. 2. Постройте график зависимости τrel = τrel (NP article). 3. Какой функцией можно аппроксимировать данную зависимость? Объясните полученный результат с физической точки зрения

ТЕМА 13. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ МИКРОКАНОНИЧЕСКОГО АНСАМБЛЯ МЕТОДОМ МОНТЕ-КАРЛО

Общая идея метода Монте-Карло для микроканонического ансамбля состоит в получении репрезентативной выборки из полного числа микросостояний и оценке с ее использованием значения вероятности Ps. Для этого можно использовать достаточно

очевидную процедуру: зафиксировать полное число частиц N и объем системы V, изменять случайным образом координаты и скорости отдельных частиц, принимать конфигурации микросостояний, имеющих заданную полную энергию. К сожалению, данная процедура оказывается весьма неэффективной, так как большинство конфигураций вообще не будут иметь требуемую полную энергию и должны быть отвергнуты.

Эффективная вычислительная процедура метода Монте-Карло для микроканонического ансамбля была предложена ранее. Она состоит в разделении исходной макроскопической системы на две подсистемы: исходную, называемую далее системой, и подсистему, состоящую из одного элемента, которую авторы назвали демоном (по аналогии с демоном Максвелла). Роль демона аналогична роли члена кинетической энергии в методе молекулярной динамики, обсуждавшегося в главе 10. Обходя элементы системы и передавая энергию, он обеспечивает изменение конфигурации системы. Если энергии, запасенной в мантии демона оказывается достаточно, он отдает энергию тому элементу системы, которому требуется энергия, для осуществления изменения конфигурации. И наоборот, если для изменения конфигурации требуется уменьшить энергию системы, энергия передается частицей демону. Единственное ограничение состоит в том, что энергия демона должна быть положительной. Описанная процедура алгоритмически выглядит следующим образом.

1. Сгенерировать начальную конфигурацию системы с заданным значением полной энергии.
2. Выбрать случайным образом частицу и произвести пробное изменение ее координат.
3. Вычислить полную энергию системы в новом состоянии.
4. Если в новом состоянии энергия системы оказывается меньше, то система отдает энергию демону и новая конфигурация принимается.
5. Если в новом состоянии энергия системы оказывается больше, то новая конфигурация принимается, если энергии демона достаточно, чтобы передать ее системе. В противном случае новая конфигурация не принимается и частица сохраняет свои старые координаты.
6. Если пробное изменение не меняет энергию системы, то новая конфигурация принимается.

Перечисленные выше действия повторяются до получения репрезентативной выборки микросостояний. Можно ожидать, что через некоторый промежуток времени система достигнет равновесного состояния, в котором у системы и демона будут некоторые средние (для каждого) значения энергии. При этом значение полной энергии системы остается постоянной. Так как демон представляет собой только одну степень свободы, в отличие от рассматриваемой статистической системы можно ожидать, что флуктуации системы будут невелики.

Применим описанный алгоритм к классическому одномерному идеальному газу, скорости частиц которого непрерывны и неограниченны, а энергия частиц не зависит от положения частиц, поэтому полная энергия частиц есть сумма кинетических энергий отдельных частиц. Для изменения конфигурации будем случайным образом изменять скорость случайно выбранной частицы.

Для реализации описанного алгоритма в MATLAB создадим два m-файла:

1) файл Init.m, содержащий описание функции, возвращающей абсолютные значения скоростей в момент времени t = 0;

2) файл Demon.m, содержащий описание функции, возвращающей усредненные по ансамблю реализаций значения: энергии демона (Edave), средней энергии системы на одну частицу (Esysave), средней скорости на одну частицу системы (Vave), среднего числа принятия (Accept).

```
% листинг файла Init.m
function z = Init(N,Esystem)
% функция, возвращающая
% абсолютные значения
% скоростей в момент
```

```
% времени t = 0
% N - число частиц системы
% Esystem - энергия системы
V=(Esystem./N).^0.5;
for i=1:N
   z(i)=V;
end;

% листинг файла Demon.m
function [Edave,Esysave,Vave,Accept] = Demon(N,Esystem,NTrial,Vel,dV)
% функция, возвращающая усредненные
% по ансамблю реализаций
% значения: энергии демона (Edave),
% средней энергии системы на одну
% частицу (Esysave), средней скорости на одну
% частицу системы (Vave),
% среднего числа принятия (Accept)

Edemon=0;
Vtot=sum(Vel);
Vcum=0;
Escum=0;
Edcum=0;
Accept=0;
for j=1:NTrial
   for i=1:N
      % случайное изменение скорости
      dv=(2*rand(1)-1)*dV;

      % случайный выбор частицы
      Ip=floor(N*rand(1)+1);

      % пробное изменение скорости
      VTrial=Vel(Ip)+dv;

      % пробное изменение энергии
      de=0.5*(VTrial.^2-Vel(Ip)^2);

      % если энергия уменьшается
      % изменение принимается
      if de<=Edemon
         Vel(Ip)=VTrial;
         Vtot=Vtot+dv;
         Accept=Accept+1;
         Edemon=Edemon-de;
         Esystem=Esystem+de;


      end;
      Edcum=Edcum+Edemon;
      Escum=Escum+Esystem;
      Vcum=Vcum+Vtot;
   end;
end;
Edave=Edcum/(NTrial*N);
Accept=Accept/(NTrial*N);
Esysave=1/N*Escum/(NTrial*N);
Vave=1/N*Vcum/(NTrial*N);
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava14_1.m:

```
% листинг файла Glava14_1.m
Esystem=40; % энергия системы
N=40; % число частиц системы
NTrial=4000; % число испытаний
dV=2*2^0.5; % максимальное значение
            % изменения скорости
Vel=InitD(N,Esystem); % задание
                      % начальных скоростей
% вычисление характеристик
% состояния идеального газа
[Edave Esysave Vave Accept]=...
    Demon(N,Esystem,NTrial,Vel,dV);
Edave
Esysave
Accept
```

Типовые задания

Задача 1. Модифицируйте функцию Demon.m, листинг которой приведен выше, так, чтобы номера частиц, скорость которых подлежит изменению (переменная Ip), выбирались последовательно. Приводит ли изменение способа выбора номера частицы к изменению средней энергии системы на частицу и средней энергии демона?

14. ТЕМА МОДЕЛИРОВАНИЕ КАНОНИЧЕСКОГО АНСАМБЛЯ МЕТОДОМ МОНТЕ-КАРЛО

Алгоритм метрополиса для канонического ансамбля

Рассмотрим систему, состоящую из N частиц, находящихся в объеме V при постоянной температуре T. Сгенерируем некоторое ограниченное число конфигураций m из полного числа возможных конфигураций M. Тогда оценка среднего значения ⟨A⟩ может быть получена из следующего выражения

$$\langle A \rangle \approx \sum_{s=1}^{m} A_s P_s = \frac{\sum_{s=1}^{m} A_s e^{\frac{-E_s}{T}}}{\sum_{s=1}^{m} e^{\frac{-E_s}{T}}}$$

E_s и A_s – полная энергия системы E и значение физической величины A в конфигурации s

Для вычисления ⟨A⟩ можно предложить достаточно очевидную процедуру, которая состоит в генерации случайной конфигурации, вычислении E_s, A_ss и произведения

$$A_s e^{\frac{-E_s}{T}}$$

и подсчете соответствующего вклада каждой конфигурации в сумму. Однако появление каждой конкретной конфигурации достаточно маловероятно, следовательно, большое число конфигураций будут давать малый вклад в суммы, поэтому, вообще говоря, такой вычислительный алгоритм также будет недостаточно эффективным.

Для устранения отмеченного недостатка можно воспользоваться методом существенной выборки для вычисления определенных интегралов методом Монте-Карло. Идея данного

метода состоит в генерации конфигурации в соответствии с функцией распределения вероятностей π_s. Так как дальнейшее усреднение проводится по m конфигурациям смещенной выборки, то для исключения данного смещения при вычислении суммы каждая конфигурация должна браться с весовым множителем $1/\pi_s$

$$\langle A \rangle \approx \frac{\sum_{s=1}^{m} A_s \frac{1}{\pi_s} e^{\frac{-E_s}{T}}}{\sum_{s=1}^{m} \frac{1}{\pi_s} e^{\frac{-E_s}{T}}}$$

Наиболее целесообразно оказывается выбрать $1/\pi_s$ в виде:

$$\pi_s = \frac{e^{\frac{-E_s}{T}}}{\sum_{s=1}^{m} e^{\frac{-E_s}{T}}}$$

при преобразовании знаменателя учтено условие нормировки

$$\sum_{s=1}^{m} e^{\frac{-E_s}{T}} = 1$$

Общая формулировка алгоритма Метрополиса:

1. Сформировать начальную конфигурацию системы.
2. Произвести случайное пробное изменение начальной конфигурации.
3. Вычислить изменение энергии системы ΔE, обусловленное произведенным пробным изменением конфигурации.
4. Если $\Delta E \leq 0$, то принять новую конфигурацию системы и осуществить переход к п. 8.
5. Если $\Delta E \geq 0$, то вычислить «вероятность перехода» $W = e^{\frac{\Delta E}{T}}$
6. Сгенерировать случайное число r с равномерным законом распределения в интервале [0, 1].
7. Если $r \leq W$, принять новую конфигурацию, в противном случае сохранить предыдущую конфигурацию.
8. Определить значения требуемых физических величин.
9. Повторить п. 2–8 для получения достаточного числа конфигураций.
10. Вычислить средние по статистически независимым друг от друга конфигурациям.

Описанный алгоритм можно трактовать как случайные блуждания. Действительно, пронумеруем различные конфигурации порядковыми номерами i = 1, 2, 3, . . . Тогда каждую конфигурацию можно считать некоторой «точкой», а вычислительный процесс в соответствии с алгоритмом Метрополиса случайным блужданием по данным точкам. Действия, выполняемые в соответствии с п. 3–7, позволяют определить условную вероятность того, что в «момент времени» прохожий будет находиться в точке i при условии, что в момент времени t он находился в точке j, то есть найти отношение P(i)/P(j). Отметим, что поскольку вычисляется отношение вероятностей, нет необходимости проводить их нормировку. Так как конфигурации генерируются с вероятностью, пропорциональной требуемой вероятности, для вычисления средних следует использовать формулу (15.9). Отметим, что данным способом оказывается возможным оценить средние значения макроскопических величин, но не нормировочный коэффициент Z

Существует строгое математическое доказательство того факта, что после достаточно большого числа шагов алгоритм Метрополиса генерирует состояния, энергия которых распределена по Больцману. Однако, следуя выбранному подходу (физическому по своей сути), мы сознательно отказываемся от математически строгого доказательства в пользу эмпирического подхода, то есть подтверждения данного вывода на примере модели идеального газа и одномерной системы спинов.

Отметим, что выбор в качестве распределения Больцмана не является единственно возможным. Оказывается, что к распределению Больцмана в асимптотическом пределе приводят и другие распределения вероятности. Можно показать, что существует единственное необходимое условие, заключающееся в том, чтобы

$$W(1 \to 2)e^{\frac{-E_1}{T}} = W(2 \to 1)e^{\frac{-E_2}{T}}$$

Это равенство называется принципом детального равновесия

На примере классического идеального газа покажем, что алгоритм Метрополиса приводит для отдельных микросостояний к распределению Больцмана. Любое микросостояние газа полностью описывается заданием скорости частиц, так как энергия идеального газа зависит только от скорости частиц. При этом скорость классической частицы является величиной непрерывной, в то время как для количественного описания системы необходимо иметь счетное число микросостояний. Следовательно, необходимо провести разбиение интервала возможных значений скоростей на конечное число достаточно малых дискретных интервалов. Например, для 20 частиц и разделении интервала возможных скоростей на 10 интервалов полное число возможных микросостояний составляет 10^{20}. Совершенно очевидно, что весьма проблематично не только пронумеровать данные состояния, но и вычислить за приемлемое время точные оценки вероятностей и средних величин. Полученная оценка подтверждает правомерность использования модели идеального газа для демонстрации асимптотических характеристик алгоритма Метрополиса.

Для реализации алгоритма Метрополиса в модели идеального газа, состоящего из одной частицы, в MATLAB создадим файл IdealGas.m, содержащий описание функции, возвращающей мгновенные значения скорости, энергии, среднее число принятия.

```
% листинг файла IdealGas.m
function [V,Es,Accept] = IdealGas(Vel,T,dVmax,NTrial)
% функция, возвращающая
% мгновенные значения скорости (V),
% энергии (E),
% среднее число принятия (Accept)
% Vel - начальная скорость
% T - температура газа
% dVmax - максимальное изменение скорости
% NTrial - число испытаний

beta=1/T;
E=Vel.^2/2; % начальная
            % энергия
Accept=0;

for i=1:NTrial
   dV=(2*rand(1)-1)*dVmax;
   Vtrial=Vel+dV; % случайное изменение
                  % скорости
   de=0.5*(Vtrial.^2-Vel.^2); % пробное
                              % изменение энергии
```

```
   if de>0
     if exp(-beta*de)>=rand(1)
        % шаг принимается
       Vel=Vtrial;
       Accept=Accept+1;
       E=E+de;
     end;
   else
      % шаг не принимается
      Vel=Vtrial;
      Accept=Accept+1;
      E=E+de;
   end;
   Es(i)=E;
   V(i)=Vel;
end;
Accept=Accept/NTrial;
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava15_1.m:

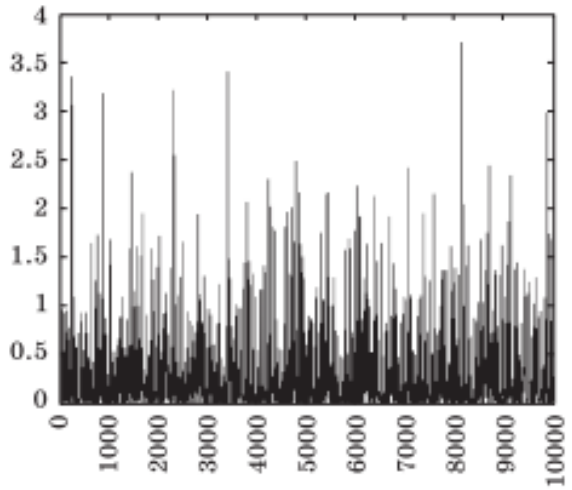
```
% листинг файла Glava15_1.m
Vel=0; % начальная скорость частицы
T=0.5; % температура газа
dVmax=4; % максимальное изменение скорости
NTrial=10^4; % число испытаний

% вычисление мгновенных
% значений скорости (V),
% энергии (E),
% среднего числа принятия (Accept)
[V Es Accept]=IdealGas(Vel,T,dVmax,NTrial);

Vmean=mean(V) % средняя скорость частицы
Esmean=mean(Es) % средняя энергия частицы
Accept % число принятия

i=1:NTrial;
% визуализация зависимости
% мгновенных значений энергии
% от времени
figure;
plot(i,Es,'k');

% вычисление и визуализация
% функции распределения  по
```

```
% энергии идеального газа,
% состоящего из одной частицы

x1=min(Es);
x2=max(Es);

Nint=100; % число интервалов гистограммы
i=1:Nint;
x(i)=x1+(x2-x1)/Nint*i;

h=hist(Es,x);
figure;
bar(x,h);
colormap white
```

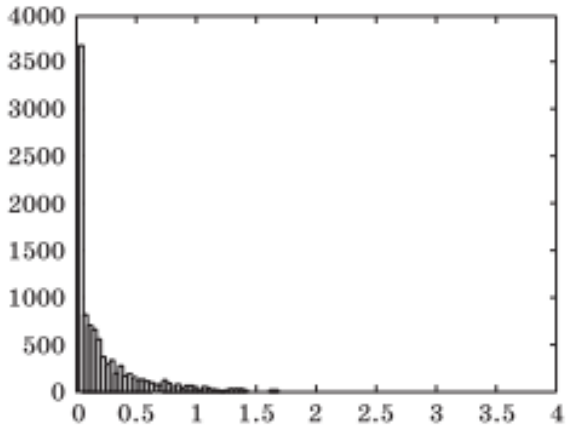
Ниже представлены результаты выполнения описанной последовательности команд.

1. Текстовые сообщения, выводимые в командное окно:

```
    Vmean =
 -7.6218e-004
  Esmean =
    0.2425
  Accept =
    0.2883
```



. Зависимость мгновенных значений энергии идеального газа, состоящего из одной частицы, от времени



Распределение случайной последовательности, представленной на предыдущем рисунке

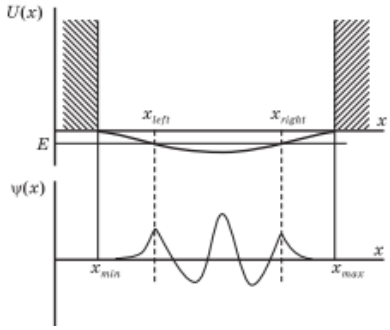
Типовые задания

Задача 1. Проведите вычисления для различных значений температуры и постройте зависимость энергии газа от температуры. Можно ли аппроксимировать данную зависимость функцией вида ax + b? Чему равен коэффициент a в данном случае? Какой физический смысл имеет коэффициент a? Объясните полученный результат с точки зрения теоремы о равнораспределении

Задача 2. Проведите вычисления для различных значений температуры и постройте

зависимости дисперсии и среднеквадратичного отклонения скорости от температуры. Выберите подходящие функции и аппроксимируйте данные зависимости методом наименьших квадратов. Объясните полученные результаты.

ТЕМА 15. МОДЕЛИРОВАНИЕ КВАНТОВЫХ СИСТЕМ



Методы численного решения стационарного уравнения Шредингера

Рассмотрим решения одномерного стационарного уравнения Шредингера частицы, движущейся в одномерном потенциале U(x). $-\frac{\hbar^2}{2m}\frac{\partial^2\varphi(x)}{\partial x^2}+U(x)\varphi(x)$

Будем при этом полагать, что его форма имеет вид, представленный на рисунке: в точках x_{min}, x_{max} потенциал становится бесконечно большим.

Это означает, что в точках x_{min}, x_{max} расположены вертикальные стенки, а между ними находится яма конечной глубины. Для удобства дальнейшего решения запишем уравнение Шредингера в виде:

$$\frac{d^2\psi}{dx^2}+k^2(x)\psi(x)=0$$

Где $k^2(x)=\frac{2m}{\hbar^2}[E-U(x)]$

С математической точки зрения задача состоит в отыскании собственных функций оператора

$$\frac{d^2}{dx^2}+k^2(x)$$

отвечающих граничным условиям

$$\psi_n(x_{\min})=\psi_n(x_{\max})=0$$

и соответствующих собственных значений энергии E_n

Так как

$$k^2(x)\geq 0$$

При $x\in x_{left},x_{right}$

$$k^2(x)<0$$

При $x\in[x_{min}x_{left}]$, $x\in[x_{right},x_{max}]$, то можно ожидать, что собственному решению данной задачи соответствует собственная функция:

1) осциллирующая в классически разрешенной области движения $E\geq U(x)$, когда $x\in[x_{left},x_{right}]$

2) экспоненциально затухающая в запрещенных областях, где $E<U(x)$, когда $x\in[x_{min}x_{left}]$, $x\in[x_{right},x_{max}]$;

3) тождественно равная нулю ($\psi\equiv 0$), когда $x\leq x_{min}$, $x\geq x_{max}$.

Так как в рассматриваемом случае все состояния частицы в потенциальной яме оказываются связанными (то есть локализованными в конечной области пространства), спектр энергий является дискретным. Так как все состояния частицы в потенциальной яме оказываются связанными (то есть локализованными в конечной области пространства), спектр энергий является дискретным

В общем случае частица, находящаяся в потенциальной яме конечных размеров

$$U(x)<0 \quad if \quad x\in[x_{min},x_{max}];$$
$$U(x)\leq \quad if \quad x\notin[x_{min},x_{max}]$$

имеет дискретный спектр при E<0 и непрерывный спектр при E≥0.

Традиционно для численного решения задачи о нахождении собственных значений уравнения Шредингера используется метод пристрелки. Идея метода пристрелки состоит в следующем. Допустим, в качестве искомого значения ищется одно из связанных состояний, поэтому в качестве пробного начального значения энергии выбираем отрицательное собственное значение.

Проинтегрируем уравнение Шредингера каким-либо известным численным методом на интервале $[x_{min}, x_{left}]$. По ходу интегрирования от x_{min} в сторону больших значений x сначала вычисляется решение $\psi + (x)$, экспоненциально нарастающее в пределах классически запрещенной области. После перехода через точку поворота x_{right}, ограничивающую слева область движения, разрешенную классической механикой, решение уравнения становится осциллирующим. Если продолжить интегрирование далее за правую точку поворотаx_{right}, то решение становится численно неустойчивым. Это обусловлено тем, что даже при точном выборе собственного значения, для которого выполняется условие $\psi + (x_{max})$, решение в области $[x_{right}, x_{max}]$всегда может содержать некоторую примесь экспоненциально растущего решения, не имеющего физического содержания.

Отмеченное обстоятельство является общим правилом: интегрирование по направлению внутрь области, запрещенной классической механикой, будет неточным. Следовательно, для каждого значения энергии более разумно вычислить еще одно решение $\psi_-(x)$, интегрируя уравнение (16.10) от x_{max} в сторону уменьшения x. Критерием правильного выбора данного значения энергии является совпадение значений функций $\psi_+(x)$ и $\psi_-(x)$ в некоторой промежуточной точке xm. (Так как функции $\psi_+(x)$ и $\psi_-(x$ являются решениями однородного уравнения Шредингера их всегда можно отнормировать так, чтобы в точке x_m выполнялось условие $\psi_+(x_m)$ и $\psi_-(x_m)$.) Обычно в качестве данной точки выбирают левую точку поворота x_{left}

Помимо совпадения значений функций в точке x_mдля обеспечения гладкости сшивки решений потребуем совпадения значений их производных $\psi'_+(x_m)$ и $\psi'_-(x_m)$ в точке x_m, находим эквивалентное условие гладкости сшивки решений:

$$f = \frac{\psi_+(x_m) \text{и } \psi_-(x_m) - [\psi_+(x_m - h) \text{ и } \psi_-(x_m + h)]}{\max|\psi|} = 0$$

Число $\max|\psi|$является масштабирующим множителем, который выбирается из условия $f \cong 1$. Если точки поворота отсутствуют, то есть $E > 0$, то в качестве x_m можно выбрать любую точку отрезка$[x_{min}, x_{max}]$. Для потенциалов, имеющих более двух точек поворота и, соответственно, три или более однородных решений, общее решение получается сшивкой отдельных кусков.

В описанном ниже документе для интегрирования дифференциального уравнения второго порядка мы используем метод Нумерова. Для получения вычислительной схемы аппроксимируем вторую производную трехточечной разностной формулой

$$\frac{y_{n+1} - 2y_n + y_{n-1}}{h^2} = y_n'' + \frac{h^2}{12} y_n^{(IV)} + O(h^4)$$

Тогда

$$y_n^{(IV)} = \frac{d^2}{dx^2}(-k^2(x)y)|_{x=x_n} = \frac{(k^2y)_{n+1} - 2(k^2y)_n + (k^2y)_{n-1}}{h^2}$$

$$\left(1 + \frac{h^2}{12}k_{n+1}^2\right)y_{n+1} - 2\left(1 - \frac{5h^2}{12}k_n^2\right)y_n + \left[1 + \frac{h^2}{12}k_{n-1}^2\right]y_{n-1} = 0$$

Разрешив lданное уравнение относительно y_{n+1} или y_{n-1}, найдем рекуррентные формулы для интегрирования уравнения Шредингера вперед или назад по x с локальной погрешностью $O(h^6)$. Отметим, что погрешность данного метода оказывается на порядок выше, чем погрешность метода Рунге–Кутта четвертого порядка. Кроме того, данный

алгоритм более эффективен, потому что значения функции $k^2(x)$вычисляются только в узлах сетки.

Для нахождения численного решения оказывается удобным провести обезразмеривание уравнения Шредингера, используя в качестве единиц измерения расстояния a — ширину потенциальной ямы, в качестве единиц измерения энергии $|U_0|$ — модуль минимального значения потенциала. В выбранных единицах измерения уравнение Шредингера принимает следующий вид:

$$\frac{d^2\psi}{dx^2} + \gamma^2\left(\tilde{E} - \tilde{U}(\tilde{x})\right)\psi = 0$$

$$\tilde{x} = \frac{x}{a}$$

$$\tilde{E} = \frac{E}{U_0}$$

$$\tilde{U}(\tilde{x}) = \frac{U(\tilde{x})}{U_0}$$

$$\gamma^2 = \frac{2ma^2U_0}{\hbar^2}$$

Таким образом, вычислительный алгоритм для нахождения собственных функций и собственных значений уравнения Шредингера реализуется следующей последовательностью действий:

1. Задать выражение, описывающее безразмерный потенциал $U(\tilde{x})$).
2. Задать значение γ^2
3. Задать пространственную сетку, на которой проводится интегрирование уравнения
4. Задать x_{min}, x_{max}.
5. Задать начальное значение энергии E_{start}.
6. Задать конечное значение энергии E_{fin}.
7. Задать шаг изменения энергии ΔE.
8. Проинтегрировать уравнение Шредингера для значения энергии E_i слева направо на отрезке [x_{min}, x_m].
9. Проинтегрировать уравнение Шредингера для значения энергии E_{i+1} справа налево на отрезке [x_{max}, x_m].
10. Вычислить значения переменной f_i для значения энергии E_i
11. Увеличить текущее значение энергии на ΔE: $E_{i+1}=E_i+\Delta E$.
12. Проинтегрировать уравнение Шредингера для значения энергии E_{i+1} слева направо на отрезке [x_{min}, x_m].
13. Проинтегрировать уравнение (16.10) для значения энергии E_{i+1} справа налево на отрезке [xmax, xm].
14. Вычислить значения переменной fi+1 для значения энергии E_{i+}
15. Сравнить знаки f_i , f_{i+1} 16. Если f_i f_{i+1} >0 и $E_{i+1}\leq E_{max}$, увеличить текущее значение энергии на ΔE и

повторить действия, описанные в п. 8–15.

17. Если f_i, f_{i+1} >0 уточнить собственное значение методом линейной интерполяции.
18. Если $E_{i+1}\leq E_{max}$, повторить действия, описанные в п. 8–18.
19. Если $E_{i+1}\leq E_{max}$, закончить вычисления.

Для реализации описанного вычислительного алгоритма в MATLAB необходимо создать три m-файла:

1) файл Num.m, содержащий описание функции, возвращающей для заданной энергии величину *f*, и значения волновой функции в узлах координатной сетки;

2) файл U.m, содержащий описание функции, возвращающей значение потенциала;

3) файл Elevel.m, содержащий описание функции, возвращающей собственные значения и собственные функции уравнения Шредингера.

Ниже приведены листинги данных файлов.

```
% листинг файла Num.m
function [f,psi]=Num(gamma,E,V,Xmin,Xmax,Ngreed)
% функция, возвращающая для заданной энергии
% величину f, вычисляемую
% в соответствии с (16.19), и
% значения волновой функции (psi)
% в узлах координатной сетки
% gamma - коэффициент, входящий
% в обезразмеренное уравнение
% Шредингера (16.23)
% E - заданное значение энергии
% V - вектор, содержащий значения потенциала
% в узлах координатной сетки
% Xmin - левая граница координатной сетки
% Xmax - правая граница координатной сетки
% Ngreed - число узлов координатной сетки

dx=(Xmax-Xmin)/Ngreed;
c=dx.^2*gamma^2/12;
  Imatch=1;
  psi(1)=0;
  psi(2)=9.99999*10^-10;

  % интегрирование уравнения Шредингера
  % справа налево
  Kim1=c*(E-V(1));
  Ki=c*(E-V(2));
  for i=2:Ngreed-1
     Kip1=c*(E-V(i+1));
     if and(Ki*Kip1<=0,Ki>0)
        Imatch=i;
        i=Ngreed;
     end;
     if i<=Ngreed-1
        psi(i+1)=(psi(i)*(2-10*Ki)-...
                  psi(i-1)*(1+Kim1))/(1+Kip1);
        if abs(psi(i+1))>=10^10
          for k=1:i+1
             psi(k)=psi(k)*9.99999*10^-6;
          end;
        end;
       Kim1=Ki;
       Ki=Kip1;
    end;
  end;
  if Imatch==1
    Imatch=Ngreed-10;
  end;

  % интегрирование уравнения Шредингера
  % слева направо
  psi_Left=psi(Imatch);
  psi(Ngreed)=0;
  psi(Ngreed-1)=9.99999*10^-10;
  Kip1=c*(E-V(Ngreed));
  Ki=c*(E-V(Ngreed-1));
  for i=Ngreed-1:-1:Imatch+1
     Kim1=c*(E-V(i-1));
     psi(i-1)=(psi(i)*(2-10*Ki)-...
               psi(i+1)*(1+Kip1))/(1+Kim1);
     if abs(psi(i))>10^10
       for k=Ngreed:-1:i
          psi(k)=psi(k)*9.99999*10^-6;
       end;
```

```
  end;
  Kip1=Ki;
  Ki=Kim1;
end;
end;
if psi_Left<0
   for i=Imatch:Ngreed
      psi(i)=-psi(i);
   end;
end;
psi1=abs(psi);
Psimax=max(psi1);

% вычисление разности между волновыми
% функциями, полученными
% интегрированием уравнения Шредингера
% слева направо и справа налево,
% в узле с номером Imatch
f=(psi_Left+psi(Imatch)-...
   (psi(Imatch-1)+...
   psi(Imatch+1)))/Psimax;

% листинг файла U.m
function z = U(x,Xmin,Xmax)
% функция, возвращающая значения потенциала
% в узлах координатной сетки
% x - вектор, содержащий координаты узлов
% Xmin - левая граница потенциала
% Xmax- правая граница потенциала

if and(Xmin<=x,x<=Xmax)
  z=-1;
else
  z=10^309;
end;

% листинг файла Elevel.m
function [EL,Psi]=Elevel(gamma,dE,V,Emax,Xmin,Xmax,Ngreed)
% функция, возвращающая
% собственные значения и собственные функции
% уравнения Шредингера
% gamma - коэффициент, входящий
% в обезразмеренное уравнение
% Шредингера (16.23)
% dE - приращение энергии
% V - вектор, содержащий значения потенциала
% в узлах координатной сетки
```

```
% Emax - максимальное значение энергии
% Xmin - левая граница координатной сетки
% Xmax - правая граница координатной сетки
% Ngreed - число узлов координатной сетки

Tolf=10^-6;
Emin=-1;
E=Emin+dE;
m=1;
Start=0;
while E<Emax
  if Start==0
    E1=E;
    [f,psi]=Num(gamma,E,V,Xmin,Xmax,Ngreed);
    E=E+dE;
    F1=f;
    Start=1;
  end;
  E2=E;
  [f,psi]=Num(gamma,E,V,Xmin,Xmax,Ngreed);
  F2=f;
  if F1*F2>0
    E1=E;
    F1=F2;
    E=E+dE;
  end;
  if F1*F2<0
  % уточнение собственного
  % значения энергии
  % и вычисление значений
  % собственной функции
  % в узлах координатной сетки
    a=(E2-E1)/(F2-F1);
    E=E1-a*F1;
    [f,psi]=Num(gamma,E,V,Xmin,Xmax,Ngreed);
    F1=F2;
    E1=E2;
    EL(m,1)=m;
    EL(m,2)=E;
    if m==1
      Psi0=psi';
    else
      Psi0=cat(2,Psi0,psi');
    end;
    m=m+1;
```

```
    E=E+dE;
  end;
end;
dx=(Xmax-Xmin)/(Ngreed-1);
N1=size(Psi0,2);
% нормировка собственных волновых функций
for i=1:N1
  S=Psi0(:,i);
  S1=S.^2;
  Norm=0;
  for j=1:Ngreed-1
    Norm=Norm+0.5*(S1(j)+S1(j+1))*dx;
  end;
  S=S./(Norm^0.5);
  if i==1
    Psi=S;
  else
    Psi=cat(2,Psi,S);
  end;
end;
```

Далее необходимо выполнить следующую последовательность команд, сохраненную нами в файле Glava16_1.m:

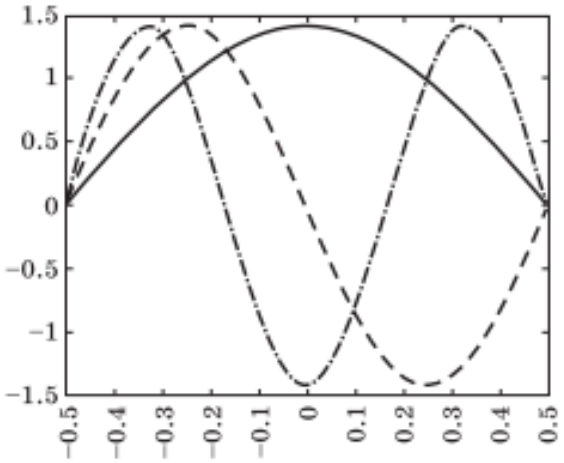
```
% листинг файла Glava16_1.m
Xmin=-0.5; % левая граница координатной сетки
Xmax=0.5; % правая граница координатной сетки
gamma=20;
Ngreed=500; % число узлов координатной сетки
% вычисление координат узлов сетки
i=1:Ngreed;
x(i)=Xmin+(Xmax-Xmin)/(Ngreed-1)*(i-1);
dE=10^-3; % приращение энергии
Emax=-0; % максимальное значение энергии
V(i)=U(x(i),Xmin,Xmax); % вычисление значения
                        % потенциала в узлах
                        % координатной сетки
% вычисление собственных значений и
% собственных функций
% уравнения Шредингера
[EL,Psi]=Elevel(gamma,dE,V,Emax,Xmin,Xmax,Ngreed);

% вывод в командном окне
% собственных значений
% уравнения Шредингера
EL
```

```
% визуализация собственных функций
% уравнения Шредингера
figure
plot(x(i),Psi(:,1),'k',...
     x(i),Psi(:,2),'--k',...
     x(i),Psi(:,3),'-.k');

figure
plot(x(i),Psi(:,4),'k',...
     x(i),Psi(:,5),'--k',...
     x(i),Psi(:,6),'-.k');
```

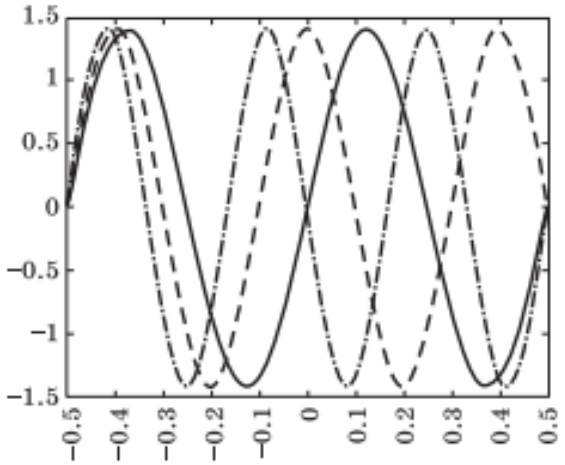
Ниже представлены результаты выполнения описанной последовательности команд.

1. Сообщения, выводимые в командное окно:

```
EL =
    1.0000   -0.9751
    2.0000   -0.9005
    3.0000   -0.7761
    4.0000   -0.6020
    5.0000   -0.3782
    6.0000   -0.1046
```



Волновые функции, соответствующие первому, второму и третьему собственным значениям энергии



Волновые функции, соответствующие четвертому, пятому и шестому собственным значениям энергии

Типовые задания

Задача 1. Проверьте условие ортогональности собственных функций уравнения Шредингера 2. Оцените точность выполнения данного условия. Исследуйте зависимость точности ортогональности собственных функций уравнения Шредингера от шага изменения энергии ΔE и числа узлов пространственной сетки Ngreed

Задача 2 Для бесконечной прямоугольной ямы шириной a удается найти аналитические выражения для собственных функций уравнения и собственных значений Шредингера

Используя безразмерные переменные, получите выражения для безразмерных значений энергии и собственных функций. Сравните численные значения энергий и волновых функций с соответствующими значениями, вычисляемыми по аналитическим выражениям. Оцените точность численных значений энергии и волновых функций. Почему точность зависит от номера собственного значения энергии?

$$
\begin{aligned}
n=1, \qquad & E_1=\frac{\hbar^2\pi^2}{2ma^2}, && \psi_1=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\cos\left(\frac{\pi\, x}{\mathrm{a}}\right),\\
n=2, \quad & E_2=4\frac{\hbar^2\pi^2}{2ma^2}=4E_1, && \psi_2=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\sin\left(\frac{2\pi\, x}{a}\right),\\
n=3, \quad & E_3=9\frac{\hbar^2\pi^2}{2ma^2}=9E_1, && \psi_3=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\cos\left(\frac{3\pi\, x}{a}\right),\\
n=4, \quad & E_4=16\frac{\hbar^2\pi^2}{2ma^2}=16E_1, && \psi_4=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\sin\left(\frac{4\pi\, x}{a}\right),\\
n=5, \quad & E_5=25\frac{\hbar^2\pi^2}{2ma^2}=25E_1, && \psi_5=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\cos\left(\frac{5\pi\, x}{\mathrm{a}}\right),\\
n=6, \quad & E_5=36\frac{\hbar^2\pi^2}{2ma^2}=36E_1, && \psi_6=\left(\frac{a}{2}\right)^{-\frac{1}{2}}\sin\left(\frac{6\pi\, x}{a}\right).
\end{aligned}
$$

СПИСОК ЛИТЕРАТУРЫ

Основная литература

1. Бахвалов Н.С., Жидков Н.П. Численные методы. Бином. 2013. 640 с.
2. Зализняк В.Е. Численные методы. Основы научных вычислений. Юрайт. 2020. 357 с.
3. Зализняк В.Е. Введение в математическое моделирование. Юрайт. 2021. 134 с.
4. Бастрон А. А., Охапкина Е. П. Практикум по численным методам в вычислительных средах MATLAB и Mathcad. РГГУ. 2019. 162 с.
5. Самарский А. А. Введение в численные методы. Лань. 2005. 288 с.

Дополнительная литература

1. Самарский А.А., Михайлов А.П. Математическое моделирование: Идеи. Методы. Примеры. Физматлит. 2005. 320 с.
2. Пименов В.Г. Численные методы в 2 частях. 2020. 112 с.
3. Самарский А.А., Гулин А.В. Численные методы. Наука. 1989. 432 с.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ИЗУЧЕНИЮ ТЕОРЕТИЧЕСКОГО МАТЕРИАЛА, ПОДГОТОВКЕ К ЛАБОРАТОРНЫМ РАБОТАМ

При подготовке к лабораторным занятиям необходимо:

- повторить материал соответствующих лекций;
- внимательно изучить описание лабораторного занятия, изложенное в практикуме;
- ответить на вопросы по содержанию лабораторного занятия, задаваемые преподавателем на предварительном опросе перед выполнением заданий.

Все лабораторные занятия предполагают выполнение заданий, решение задач.

В результате выполнения заданий достигаются следующие результаты обучения:

- развитие мышления и творческих способностей студента,
- приобретение и закрепление навыков самостоятельной работы,
- приобретение и закрепление навыков оформления и составления письменных заданий.

Письменный отчет должен содержать следующие примерные компоненты.

Отчет по лабораторной работе №
«Название работы»
выполненной студентом _____ФИО_____ ____ курса, института _____
группа_____ «___»_____201 г.
Цель работы:…….

Задание 1. Формулировка задания
Таблицы
Графики
Вычисления
Расчет погрешностей

Задание 2. Формулировка задания
Таблицы
Графики
Вычисления
Расчет погрешностей
…..
Вывод:

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ

Методические указания для обучающихся по организации и проведению самостоятельной работы

Направление подготовки 03.03.02 Физика
Направленность (профиль)
«Фундаментальная и прикладная физика»

Ставрополь
2026

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Самостоятельная работа – планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студента).

Самостоятельная работа студента является важным видом учебной и научной деятельности студента. Федеральным государственным образовательным стандартом предусматривается значительный объем времени из общей трудоемкости дисциплины на самостоятельную работу. В связи с этим, обучение включает в себя две, практически одинаковые по объему и взаимовлиянию части – процесса обучения и процесса самообучения. Поэтому самостоятельная работа должна стать эффективной и целенаправленной работой студента.

Концепцией модернизации российского образования определены основные задачи профессионального образования – подготовка квалифицированного работника соответствующего уровня и профиля, конкурентоспособного на рынке труда, компетентного, ответственного, свободно владеющего своей профессией и ориентированного в смежных областях деятельности, способного к эффективной работе по направлению 03.03.02 Физика на уровне мировых стандартов, готового к постоянному профессиональному росту, социальной и профессиональной мобильности.

Решение этих задач невозможно без повышения роли самостоятельной работы студентов над учебным материалом, усиления ответственности преподавателей за развитие навыков самостоятельной работы, за стимулирование профессионального роста студентов, воспитание творческой активности и инициативы.

Самостоятельная работа студента предполагает, прежде всего, его научно-исследовательскую и практическую деятельность, поскольку именно эти виды учебной работы в первую очередь готовят к самостоятельному выполнению профессиональных задач. Конечной целью самостоятельной работы является организация самостоятельной познавательной деятельности, формирование умения самостоятельно анализировать информацию, выделять главное и второстепенное.

Основными формами самостоятельной работы студентов по дисциплине «Компьютерное моделирование физических процессов» являются:

– подготовка к лабораторной работе;

– самостоятельное изучение литературы;

Лабораторная работа – подразумевает изучение определенного физического процесса на практике, используя при этом методы, предварительно изученные на лекциях, выбор наиболее оптимального приема выполнения замеров и исследования, которые обеспечивает наиболее точный результат, определение фактического результата и его сравнение с теоретическими данными, описанными в учебнике согласно выбранной тематике, обнаружение причин полученного несоответствия и грамотное изложение их в отчете лабораторной работы, грамотное оформление выводов согласно требованиям методички.

В результате изучения дисциплины у обучающихся формируются следующие компетенции (в соответствии с образовательным стандартом):

ИД-1$_{ПК-3}$ понимает стратегию и тактику проектной деятельности как целенаправленной, системной, профессиональной деятельности

ИД-2$_{ПК-3}$умеет применять приобретенные знания, умения и навыки в своей проектной деятельности;

ИД-1$_{ПК-5}$ ориентируется в современных тенденциях развития цифровых технологий, выбирает технологии или программные средства для решения поставленных задач.

ИД-2$_{ПК-5}$ умеет обрабатывать и анализировать полученные данные с помощью современных информационных технологий;

ИД-3$_{ПК-5}$ обладает готовностью к участию в подготовке проектной документации, в том числе с использованием средств автоматизированного проектирования и вычислительных программных комплексов, современного программного обеспечения, в том числе текстовых редакторов и графических программы;

ИД-4$_{ПК-5}$ владеет современным программным обеспечением, в том числе текстовыми редакторами и графическими программами, средствами подготовки обзоров, отзывов, отчетов, заключений

ПЛАН-ГРАФИК ВЫПОЛНЕНИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Вид деятельности студентов	Итоговый продукт самостоятельной работы	Средства и технологии оценки	Время сдачи
Подготовка к лабораторной работе по теме Моделирование относительных движений в классической механике	Отчет	Собеседование	Лабораторная работа № 1
Подготовка к лабораторной работе по теме Физические процессы, описываемые дифференциальными уравнениями первого порядка	Отчет	Собеседование	Лабораторная работа № 2
Подготовка к лабораторной работе по теме Динамика материальной точки	Отчет	Собеседование	Лабораторная работа № 3
Подготовка к лабораторной работе по теме Задача Кеплера	Отчет	Собеседование	Лабораторная работа № 4
Подготовка к лабораторной работе по теме Моделирование статических электрических и магнитных полей	Отчет	Собеседование	Лабораторная работа № 6
Подготовка к лабораторной работе по теме Моделирование движения электрических зарядов в постоянных электрических и магнитных полях	Отчет	Собеседование	Лабораторная работа № 8
Подготовка к лабораторной работе по теме Фурье анализ непрерывных и дискретных функций	Отчет	Собеседование	Лабораторная работа № 10
Подготовка к лабораторной работе по теме Моделирование колебательных процессов	Отчет	Собеседование	Лабораторная работа № 12
Подготовка к лабораторной работе по теме	Отчет	Собеседование	Лабораторная работа № 14

Вид деятельности студентов	Итоговый продукт самостоятельной работы	Средства и технологии оценки	Время сдачи
моделирование систем, состоящих из большого числа частиц, методом броуновской динамики			
Подготовка к лабораторной работе по теме Методы Монте-Карло	Отчет	Собеседование	Лабораторная работа № 16
Подготовка к лабораторной работе по теме Случайные блуждания	Отчет	Собеседование	Лабораторная работа № 18
Подготовка к лабораторной работе по теме Моделирование статистической системы в процессе релаксации и состоянии равновесия	Отчет	Собеседование	Лабораторная работа № 20
Подготовка к лабораторной работе по теме Компьютерное моделирование микроканонического ансамбля методом Монте-Карло	Отчет	Собеседование	Лабораторная работа № 22
Подготовка к лабораторной работе по теме Моделирование канонического ансамбля методом Монте-Карло	Отчет	Собеседование	Лабораторная работа № 24
Подготовка к лабораторной работе по теме Моделирование квантовых систем	Отчет	Собеседование	Лабораторная работа № 27
Самостоятельное изучение литературы	Конспект	Собеседование	