

С. Ю. Кулабухов

Издательство «Легион», г. Ростов-на-Дону, Россия

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ НА УРОКАХ ИНФОРМАТИКИ С ПОМОЩЬЮ ЧИСЛЕННОГО РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ*

Аннотация

В статье рассмотрен предлагаемый автором фрагмент углубленного курса информатики в физико-математической школе. Он основан на использовании дифференциальных уравнений для моделирования сложных физических процессов. Рассмотрены три модели: математический пружинный маятник, орбитальное движение спутника вокруг планеты и движение тела в атмосфере с учетом сопротивления воздуха. Поскольку в углубленном курсе математики физико-математической школы численные методы решения дифференциальных уравнений не изучаются, то в статье предлагается использовать самый простой метод — метод Эйлера. С его помощью осуществляется численное решение получающихся в каждом случае дифференциальных уравнений. Так как возникающие в каждой модели дифференциальные уравнения имеют второй порядок, то для применения метода Эйлера необходимо предварительно с помощью введения новых переменных понижать порядок дифференциальных уравнений. При этом вместо одного дифференциального уравнения второго порядка возникает система двух уравнений первого порядка. Для каждой построенной модели приведена ее реализация на языке программирования PascalABC.NET.

Ключевые слова: уроки информатики в физико-математической школе, математическое моделирование, метод Эйлера численного решения дифференциального уравнения, язык программирования PascalABC.NET.

DOI: 10.32517/2221-1993-2021-20-2-14-21

1. Введение

В данной статье рассматривается фрагмент углубленного курса информатики в физико-математической школе, посвященный моделированию физических процессов с помощью численного решения дифференциальных уравнений. В качестве примера приводятся три математические модели физических процессов. Эти процессы хорошо известны школьникам из курса физики. Для численного моделирования используется самый простой метод численного решения дифференциальных уравнений — метод Эйлера. Целью занятий является составление учениками компьютерной программы, которая показывает анимацию соответствующего процесса.

Естественно, что подобный цикл уроков имеет смысл проводить только после того, как школьники познакомятся с понятием дифференциального уравнения в курсе математики (см., например, [2, 6]). Необходимыми знаниями из физики учащиеся к этому моменту уже обладают. Учитывая, что в углубленном курсе математики численные решения дифференциальных уравнений, как правило, не изучаются, необходимо отдельное занятие посвятить знакомству с методом Эйлера.

Наиболее простые модели, основанные на решении дифференциальных уравнений первого порядка вида $y' = f(x, y)$, не используются в курсе. Они описывают неэффективные с точки зрения визуализации процессы,

* Материалы к статье можно скачать на сайте ИНФО: http://infojournal.ru/journals/school/school_02-2021/

Контактная информация

Кулабухов Сергей Юрьевич, канд. физ.-мат. наук, зам. ген. директора по научно-методической работе, издательство «Легион», г. Ростов-на-Дону, Россия; адрес: 344082, Россия, г. Ростов-на-Дону, ул. Согласия, д. 7; e-mail: kulabuhov@list.ru

S. Yu. Kulabukhov

Legion Publishing House, Rostov-on-Don, Russia

MATHEMATICAL MODELING IN INFORMATICS LESSONS USING NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS

Abstract

The article considers the author's fragment of an in-depth course of informatics in a physics and mathematics school. It is based on the use of differential equations to model complex physical processes. Three models are considered: a mathematical spring pendulum, the orbital motion of the satellite around the planet and the movement of the body in the atmosphere taking into account air resistance. Since in the in-depth course of mathematics of the physics and mathematics school, numerical methods for solving differential equations are not studied, the article proposes to use the simplest method — the Euler method. Using this method, the resulting differential equations in each case are numerically solved. Since differential equations arising in each model have a second order, in order to apply the Euler method, it is necessary to reduce the order of differential equations by introducing new variables. In this case, instead of one differential equation of the second order, a system of two equations of the first order arises. For each built model, its implementation in the PascalABC.NET programming language is given.

Keywords: informatics lessons in physics and mathematics school, mathematical modeling, Euler method for numerical solution of differential equation, PascalABC.NET programming language.

такие, например, как радиоактивный распад, остывание тела или жидкости и т. д.

Аналитические решения представленных ниже задач хорошо известны, но мы не используем их для моделирования по следующим причинам:

- а) необходимо познакомить учащихся с методом численного решения дифференциальных уравнений, который они не изучают в курсе «Алгебра и начала математического анализа»;
- б) аналитическое решение некоторых из приведенных дифференциальных уравнений выходит за рамки программы физико-математической школы;
- в) зачастую в задачах моделирования нецелесообразно использовать аналитические решения, так как расчеты по готовым формулам требуют вычислительных ресурсов больше, чем при получении численного решения.

Отметим, что приведенные примеры программ выполнены в очень минималистичной манере: упрощенная графика, ввод начальных данных непосредственно в тексте программы, отсутствие проверки входных данных и наличия необходимых графических файлов и т. п. Все это преследует одну цель — продемонстрировать *принципиальную возможность* для подготовленного школьника решать такие задачи, не углубляясь в излишнюю детализацию.

Следует иметь в виду, что предлагаемый материал не привязан к конкретному языку программирования. Например, две из предложенных моделей и их реализация на языке программирования Visual Basic были рассмотрены в [5].

2. Метод Эйлера

Метод Эйлера предназначен для численного решения определенной задачи Коши, а конкретнее, для поиска частного решения дифференциального уравнения вида $y' = f(x, y)$ при начальных условиях $y_0 = y(x_0)$, где функция $f(x, y)$ удовлетворяет некоторым условиям (см., например, [1, 4, 11]). Частное решение $y = y(x)$, проходящее через точку (x_0, y_0) , заменяется ломаной, каждое звено которой параллельно касательной к графику $y = y(x)$ в соответствующей граничной точке (рис. 1).

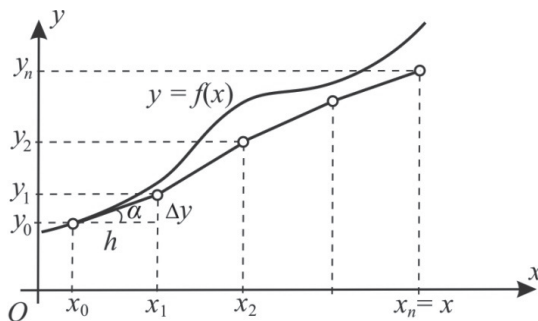


Рис. 1

Чтобы найти приближенное значение $y(x)$, полагаем $x = x_n$ и отрезок $[x_0, x_n]$ разбиваем на n равных отрезков длины:

$$h = x_{i+1} - x_i = \frac{x_n - x_0}{n} \quad (h \text{ называют шагом вычислений}).$$

Для нахождения y_1 заменяем на отрезке $[x_0, x_1]$ интегральную кривую отрезком касательной в точке (x_0, y_0) :

$$y_1 = y_0 + \Delta y = y_0 + h \cdot \operatorname{tg} \alpha = y_0 + h \cdot y'(x_0) = y_0 + h \cdot f(x_0, y_0).$$

Действуя аналогично, получим схему вычислений:

$$y_{i+1} = y_i + h \cdot f(x_i, y_i), \quad i = 0, \dots, n-1. \quad (1)$$

В результате вычислений по этой схеме получается число y_n , которое и является приближенным значением $y(x)$.

Ясно, что с уменьшением шага вычислений h ломаные Эйлера приближаются к графику искомой функции $y = y(x)$. Поэтому с уменьшением шага h метод Эйлера дает все более точное значение искомого решения в точке x .

3. Математический пружинный маятник

Такой маятник изучается на уроках физики. Модель его движения понятна школьникам из опыта. Цель занятия — построить математическую модель и написать программу, которая моделирует движение математического пружинного маятника.

На тело вдоль оси Ox действует сила натяжения пружины, направленная к началу координат и пропорциональная расстоянию от тела до точки O . Пренебрегаем силой тяжести и силой трения. Траекторию движения тела $x = x(t)$ можно найти из закона Гука и второго закона Ньютона, которые дают дифференциальное уравнение второго порядка:

$$mx'' = -kx,$$

где m — масса тела, k — коэффициент жесткости пружины.

Начальные условия задачи Коши:

$x_0 = x(0)$ — начальная координата;

$v_0 = x'(0)$ — начальная скорость тела.

Чтобы получить численное решение методом Эйлера, необходимо понизить порядок дифференциального уравнения. Это можно сделать с помощью новой переменной, имеющей очевидный физический смысл, $v = v(t)$ — скорость тела. Получим систему двух дифференциальных уравнений первого порядка, каждое из которых можно численно решить методом Эйлера:

$$\begin{cases} x' = v, \\ v' = -\frac{k}{m}x, \end{cases} \quad (2)$$

$x_0 = x(0)$, $v_0 = v(0)$ — начальные условия.

Формулы метода Эйлера (1) для каждого уравнения примут вид:

$$\begin{cases} x_{i+1} = x_i + v_i \Delta t, \\ v_{i+1} = v_i - \frac{k}{m}x_i \Delta t, \end{cases} \quad i = 0, \dots, n-1.$$

Здесь $\Delta t = t_{k+1} - t_k$ — шаг вычислений, который задается как константа. Чем он меньше, тем точнее будут найденные на последнем шаге вычислений значения x_n и v_n , которые и есть искомые $x = x(t)$ и $v = v(t)$ — приближенные значения координаты и скорости тела в момент времени t .

Так как мы полагаем $t_0 = 0$, то из того, что $\Delta t = \frac{t_n - t_0}{n}$, получим:

$$n = \frac{t_n}{\Delta t} = \frac{t}{\Delta t}.$$

В качестве примера приводится программа (листинг 1) на языке программирования PascalABC.NET (см. [3, 7–10]), в которой приведена реализация процедуры OscX(). Программа численно решает задачу Коши (2). Начальные значения x_0 и v_0 передаются по ссылке. После окончания работы процедуры в этих переменных

находятся нужные нам значения $x = x(t)$ и $v = v(t)$. Это позволяет легко организовать анимацию движения тела, для чего в программе вызываем эту процедуру через фиксированные промежутки времени и используем получаемые значения $x(t)$ для отображения положения тела в реальном масштабе времени.

На рисунке 2 приводится вид окна программы во время ее работы, когда шарик колеблется около положения равновесия. (Все представленные в статье программы можно скачать на сайте ИНФО.)

```
uses GraphABC, ABCObjects;

const interval_ms = 5;

procedure OscX(var x_0, v_0: real; t, k, m: real);
const dt = 0.0001;
var
  n: integer;
  x, v: real;
begin
  n := round(t / dt);
  k := k / m;
  for var i := 0 to n - 1 do
  begin
    x := x_0 + v_0 * dt;
    v := v_0 - k * x_0 * dt;
    x_0 := x;
    v_0 := v;
  end;
end;

var
  x_0, v_0, m, k: real;
  Ball: PictureABC := new PictureABC(0, 40, 'ball.jpg');

begin
  //Начальные данные: координата (м), скорость (м/с), масса шарика (кг),
  //жесткость пружины (Н/м)
  x_0 := 80;
  v_0 := 0;
  m := 0.1;
  k := 0.4;

  //Создание окна программы
  Window.Title := 'Пружинный маятник';
  Window.IsFixedSize := true;

  //Создание оси координат
  Window.Load('axis.bmp');

  //Основной цикл - расчет движения шарика и изменение его координаты
  while true do
  begin
    OscX(x_0, v_0, interval_ms/1000, k, m);
    Ball.Left := Round((x_0 + 100) * 3.625) + 60;
    Sleep(interval_ms);
  end;
end.
```

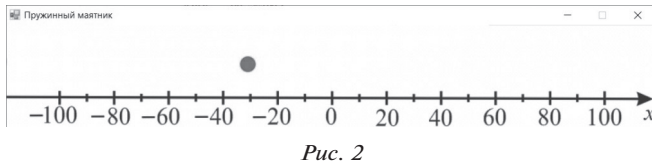


Рис. 2

4. Движение по орбите

Цель занятия — построить математическую модель и написать программу, которая моделирует движение спутника вокруг Земли.

Рассматриваем частный случай задачи двух тел, когда масса одного тела M много больше массы второго тела m . При этом тело массой M будет неподвижно. В его центре расположим центр O системы координат xOy . Траекторию движения тела массой m можно описать с помощью второго закона Ньютона системой дифференциальных уравнений для каждой координаты тела $x = x(t)$ и $y = y(t)$:

$$\begin{cases} mx'' = F_x, \\ my'' = F_y, \end{cases}$$

где F_x и F_y — компоненты вектора силы всемирного тяготения $F = G \frac{mM}{R^2}$ (G — постоянная всемирного тяготения, R — расстояние между телами).

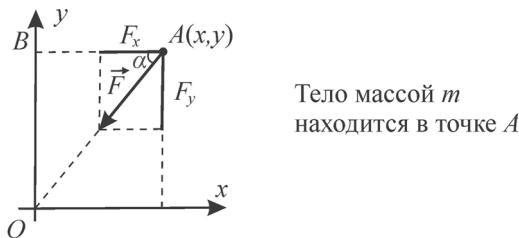


Рис. 3

Из рисунка 3 видно, что:

$$\begin{aligned} F_x &= -F \cos \alpha = -F \cdot \frac{AB}{AO} = -F \frac{x}{\sqrt{x^2 + y^2}} = \\ &= -G \frac{mM}{x^2 + y^2} \cdot \frac{x}{\sqrt{x^2 + y^2}} = -GmM \frac{x}{(x^2 + y^2)^{\frac{3}{2}}}, \end{aligned}$$

$$\begin{aligned} F_y &= -F \sin \alpha = -F \cdot \frac{BO}{AO} = -F \frac{y}{\sqrt{x^2 + y^2}} = \\ &= -G \frac{mM}{x^2 + y^2} \cdot \frac{y}{\sqrt{x^2 + y^2}} = -GmM \frac{y}{(x^2 + y^2)^{\frac{3}{2}}}. \end{aligned}$$

Получим систему:

$$\begin{cases} x'' = -GM \frac{x}{(x^2 + y^2)^{\frac{3}{2}}}, \\ y'' = -GM \frac{y}{(x^2 + y^2)^{\frac{3}{2}}}. \end{cases}$$

Чтобы применить метод Эйлера, понизим степень каждого уравнения с помощью дополнительных переменных $v_x = v_x(t)$, $v_y = v_y(t)$ (физический смысл — компоненты вектора скорости) и получим систему четырех дифференциальных уравнений первого порядка и соответствующую задачу Коши:

$$\begin{cases} x' = v_x, \\ v'_x = -GM \frac{x}{(x^2 + y^2)^{\frac{3}{2}}}, \\ y' = v_y, \\ v'_y = -GM \frac{y}{(x^2 + y^2)^{\frac{3}{2}}}, \end{cases} \quad \begin{cases} x_0 = x(0), \\ v_{x0} = v_x(0), \\ y_0 = y(0), \\ v_{y0} = v_y(0). \end{cases} \quad (3)$$

Формулы метода Эйлера (1) примут вид:

$$\begin{cases} x_{i+1} = x_i + v_{x,i} \Delta t, \\ v_{x,i+1} = v_{x,i} - GM \frac{x_i}{(x_i^2 + y_i^2)^{\frac{3}{2}}} \Delta t, \\ y_{i+1} = y_i + v_{y,i} \Delta t, \\ v_{y,i+1} = v_{y,i} - GM \frac{y_i}{(x_i^2 + y_i^2)^{\frac{3}{2}}} \Delta t, \end{cases} \quad i = 0, \dots, n-1.$$

В листинге 2 приводится программа на языке программирования PascalABC.NET, в которой приведена реализация процедуры SatelliteXY(). Программа численно решает задачу Коши (3).

```
uses GraphABC, ABCObjects;

const
  interval_sec = 5;
  Gamma = 6.67E-11; // постоянная всемирного тяготения
  R = 6400000; // радиус Земли (м)
  M = 6E+24; // масса Земли (кг)
  timeAccelerator = 10000; // коэффициент "ускорения" времени

procedure SatelliteXY(var x_0, y_0, Vx_0, Vy_0: real; t, mass: real);
const dt = 0.001;
var
  n: integer;
  x, y, Vx, Vy, p: real;
```

```

begin
  n := Round(t / dt);
  mass := Gamma * M;
  for var i := 0 to n - 1 do
    begin
      p := power(x_0 * x_0 + y_0 * y_0, 1.5);
      x := x_0 + Vx_0 * dt;
      Vx := Vx_0 - mass * x_0 * dt / p;
      y := y_0 + Vy_0 * dt;
      Vy := Vy_0 - mass * y_0 * dt / p;
      x_0 := x;
      y_0 := y;
      Vx_0 := Vx;
      Vy_0 := Vy;
    end;
  end;

var
  x_0, y_0, Vx_0, Vy_0, sc: real;
  Sat, Earth: PictureABC;

begin
  //Начальные координаты (м) и компоненты вектора скорости (м/с)
  x_0 := 0;
  y_0 := 1.2 * R;
  Vx_0 := 9000;
  Vy_0 := 1000;

  //Создание окна программы
  Window.Title := 'Движение спутника вокруг планеты';
  SetWindowSize(800, 600);
  Window.IsFixedSize := true;

  //Изображение "Земли" в центре
  Earth := new PictureABC(350, 250, 'earth.bmp');
  Earth.Transparent := true;
  Earth.Left := (Window.Width - Earth.Width) div 2;
  Earth.Top := (Window.Height - Earth.Height) div 2;

  //Создание объекта спутник
  Sat := new PictureABC(350, 250, 'satellite.jpg');

  sc := Earth.Width / 2 / R; // масштабный коэффициент: диаметр "Земли" = 2R

  //Основной цикл - расчет движения спутника и изменение его координат
  while true do
    begin
      SatelliteXY(x_0, y_0, Vx_0, Vy_0, interval_sec, M);
      Sat.Left := Round(Window.Width div 2 + x_0 * sc - Sat.Width div 2);
      Sat.Top := Round(Window.Height div 2 - y_0 * sc - Sat.Height div 2);
      sleep(Round(interval_sec * 1000 / timeAccelerator));
    end;
  end.

```

Листинг 2 (окончание)

Во время работы этой программы на экране воспроизводится движение спутника Земли. Изменяя начальные координаты и скорость спутника, можно приблизительно оценить основные законы орбитального движения (первая и вторая космические скорости, первый и второй законы Кеплера и т. д.).

5. Движение тела в атмосфере с учетом сопротивления воздуха

В школьном курсе физики рассматривается задача исследования траектории движения тела массой m , брошенного под углом α к горизонту с начальной скоростью

v_0 . Эта задача решается в предположении отсутствия сопротивления воздуха. Траектория движения тела является параболической. Построим теперь модель такого движения с учетом сопротивления воздуха. Исследование этой модели позволит увидеть отличие реальной траектории движения тела в атмосфере от параболы.

Предположим, что тело массой m бросают под углом α к горизонту с начальной скоростью v_0 . Наша цель — написать программу, которая моделирует движение тела и рисует его траекторию. При этом будет учитываться сила сопротивления воздуха, которая пропорциональна скорости тела и направлена противоположно ей. Начало системы координат xOy будет в правом нижнем углу окна программы, оси направлены как обычно: Ox — вправо, Oy — вверх.

Траекторию движения тела массой m можно описать с помощью второго закона Ньютона системой дифференциальных уравнений для каждой его координаты $x = x(t)$ и $y = y(t)$:

$$\begin{cases} mx'' = F_x, \\ my'' = F_y, \end{cases}$$

где F_x и F_y — компоненты вектора равнодействующей силы, действующей на тело. На рисунке 4 видно, что на движущееся в атмосфере тело действуют две силы: сила тяжести $\vec{F} = m\vec{g}$ и сила сопротивления воздуха $\vec{F}_c = -\beta\vec{v}$, где g — ускорение свободного падения, β — коэффициент сопротивления воздуха, $\vec{v}$ — вектор скорости тела.

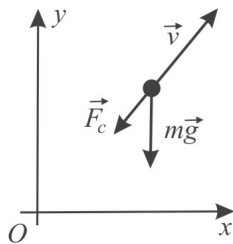


Рис. 4

Получим систему:

$$\begin{cases} x'' = -\frac{\beta}{m}v_x, \\ y'' = -g - \frac{\beta}{m}v_y. \end{cases}$$

Чтобы применить метод Эйлера, понизим степень каждого уравнения и получим систему четырех дифференциальных уравнений первого порядка и соответствующую задачу Коши:

$$\begin{cases} x' = v_x, \\ v'_x = -\frac{\beta}{m}v_x, \\ y' = v_y, \\ v'_y = -g - \frac{\beta}{m}v_y, \end{cases} \quad \begin{cases} x_0 = x(0), \\ v_{x0} = v_x(0) = v_0 \cos \alpha, \\ y_0 = y(0), \\ v_{y0} = v_y(0) = v_0 \sin \alpha. \end{cases} \quad (4)$$

Формулы метода Эйлера (1) примут вид:

$$\begin{cases} x_{i+1} = x_i + v_{x,i}\Delta t, \\ v_{x,i+1} = v_{x,i} - \frac{\beta}{m}v_{x,i}\Delta t, \\ y_{i+1} = y_i + v_{y,i}\Delta t, \\ v_{y,i+1} = v_{y,i} - \left(g + \frac{\beta}{m}v_{y,i}\right)\Delta t, \end{cases} \quad i = 0, \dots, n-1.$$

В листинге 3 приводится программа на языке программирования PascalABC.NET, которая реализует описанную модель.

В этой программе процедура StoneXY() численно решает задачу Коши (4).

На рисунке 5 приводится вид окна программы во время ее работы.

Заметим, что если в программе, представленной в листинге 3, положить коэффициент сопротивления воздуха равным нулю, то траекторией тела будет парабола, что лишний раз подтверждает адекватность математической модели.

```
uses GraphABC, ABCObjects;

const
  interval_sec = 0.01;
  g = 9.81;

procedure StoneXY(var x_0, y_0, Vx_0, Vy_0 : real; t, m, beta : real);
const dt = 0.001;
var
  n: integer;
  x, y, Vx, Vy: real;
begin
  n := Round(t / dt);
  beta := beta / m;
  for var i := 0 to n - 1 do
    begin
      x := x_0 + Vx_0 * dt;
      Vx := Vx_0 - beta * Vx_0 * dt;
```

```

    y := y_0 + Vy_0 * dt;
    Vy := Vy_0 - (g + beta * Vy_0) * dt;
    x_0 := x;
    y_0 := y;
    Vx_0 := Vx;
    Vy_0 := Vy;
end;
end;

var
    x_0, y_0, Vx_0, Vy_0, V_0, alpha, beta, m, sc: real;
    x, y: integer; // "экранные" координаты камня
    Stone: PictureABC;

begin
    // Начальные данные: координаты (м), скорость (м/с), масса камня (кг),
    // угол наклона к горизонту (градусы), коэффициент сопротивления воздуха
    x_0 := 0;
    y_0 := 0;
    V_0 := 50;
    m := 0.1;
    alpha := 60;
    beta := 0.03;

    // Определение начальных компонент вектора скорости
    alpha := alpha * Pi / 180; // перевод градусов в радианы
    Vx_0 := V_0 * cos(alpha);
    Vy_0 := V_0 * sin(alpha);

    // Создание окна программы
    Window.Title := 'Бросок камня под углом к горизонту';
    SetWindowSize(800, 600);
    Window.IsFixedSize := true;

    // Создание объекта камень
    Stone := new PictureABC(0, 550, 'stone.jpg');

    sc := 10; // Масштабный коэффициент: 1 м = 10 px

    // Основной цикл - расчет движения камня, изменение его координат, траектория
    while y_0 >= 0 do
        begin
            StoneXY(x_0, y_0, Vx_0, Vy_0, interval_sec, m, beta);
            x := Round(x_0 * sc);
            y := Round(Window.Height - y_0 * sc);
            Stone.Left := x - Stone.Width div 2;
            Stone.Top := y - Stone.Height div 2;
            SetPixel(x, y, clRed);
            sleep(Round(interval_sec * 1000));
        end;
    end.
end.

```

Листинг 3 (окончание)

Построенную модель можно применить и при написании других программ. Например, программы для моделирования прыжка с парашютом (открытию парашюта соответствует резкое увеличение коэффициента сопротивления воздуха), стрельбы из лука, артиллерийской стрельбы, сбрасывания бомбы с самолета и т. п.

6. Заключение

Рассмотренный материал является примером осуществления межпредметных связей в курсе информатики физико-математической школы. Он позволяет на практике познакомить школьников с основными этапа-

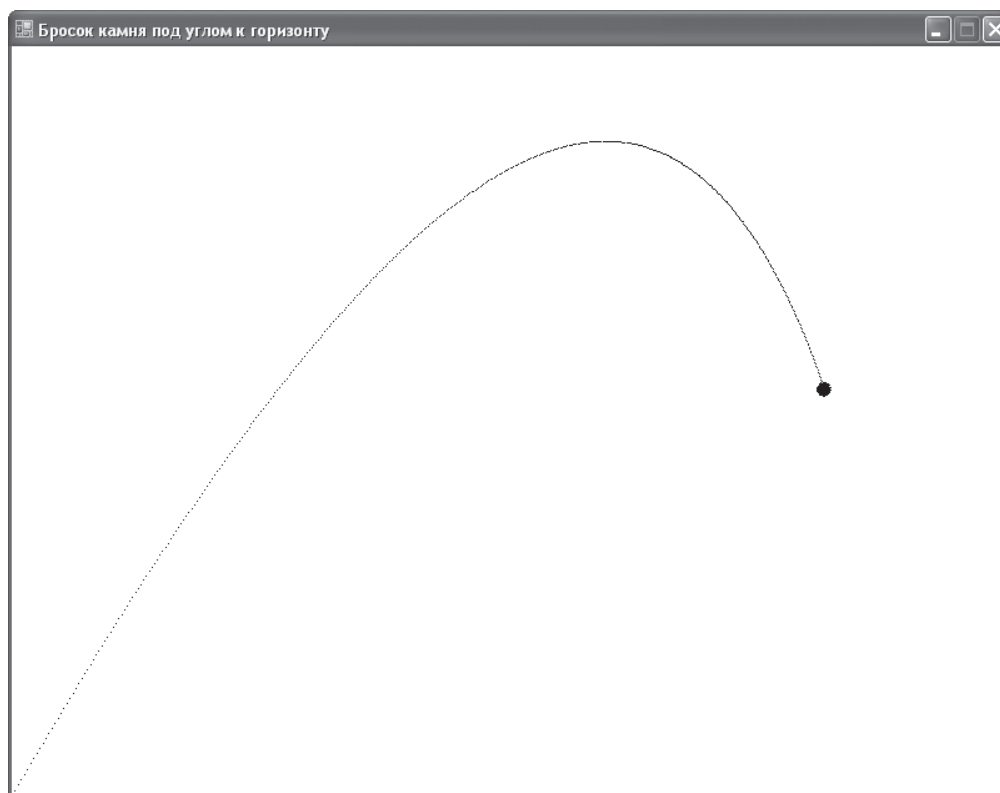


Рис. 5

ми математического моделирования — от постановки задачи до ее реализации в виде компьютерной программы и проверки построенной модели на адекватность. Кроме того, учащиеся знакомятся с одним из численных методов решения дифференциальных уравнений.

Список использованных источников

1. Бахвалов Н. С., Жидков Н. П., Кобельков Г. М. Численные методы. 3-е изд., перераб. и доп. М.: БИНОМ. Лаборатория знаний, 2003. 632 с.
2. Виленкин Н. Я., Ивашев-Мусатов О. С., Шварцбург С. И. Алгебра и начала математического анализа. 11 класс: учебник для учащихся общеобразовательных организаций (углубленный уровень). 18-е изд., стер. М.: Мнемозина, 2014. 312 с.
3. Долинер Л. И. Основы программирования в среде PascalABC.NET: учебное пособие. Екатеринбург: Изд-во Урал. ун-та, 2014. 128 с.
4. Зорич В. А. Математический анализ. Ч. I. Изд. 4-е, испр. М.: МЦНМО, 2002. 664 с.
5. Кулабухов С. Ю. Математическое моделирование на уроках информатики с использованием метода Эйлера — численного решения дифференциальных уравнений // Инфор-

матизация образования и методика электронного обучения: цифровые технологии в образовании. Труды IV Международной научной конференции (Красноярск, 6–9 октября 2020 г.) / под общ. ред. М. В. Носкова. Красноярск: СФУ, 2020. С. 62–66.

6. Никольский С. М., Потапов М. К., Решетников Н. Н., Шевкин А. В. Алгебра и начала математического анализа. 11 класс: учебник для общеобразовательных учреждений: базовый и профильный уровни. 8-е изд. М.: Просвещение, 2009. 464 с.

7. Осипов А. В. PascalABC.NET: Введение в современное программирование. Ростов-на-Дону, 2019. 572 с. <http://pascalabc.net/downloads/OsipovBook/КнигаДляСайта.pdf>

8. Осипов А. В. PascalABC.NET: выбор школьника. Ч. 1. 2-е изд., испр. и доп. Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2020. 148 с. <http://pascalabc.net/downloads/OsipovBook/StudentChoice.pdf>

9. Осипов А. В. PascalABC.NET: выбор школьника. Ч. 2. Ростов-на-Дону, 2020. 179 с. <http://pascalabc.net/downloads/OsipovBook/StudentChoice2.pdf>

10. Система программирования PascalABC.NET. <http://pascalabc.net>

11. Эльсгольц Л. Э. Дифференциальные уравнения и вариационное исчисление. М.: 1969. 424 с.