Численные методы решения обыкновенных дифференциальных уравнений и их систем

Численное решение задачи Коши для обыкновенных дифференциальных уравнений.

Обыкновенным дифференциальным уравнением (ОДУ) n-го порядка называется уравнение вида

$$F(x, y, y', ..., y^{(n)}) = 0,$$

где x – независимая переменная, $y(x)$ – искомая функция, $y', ..., y^{(n)}$ – ее производные, F – заданная функция $(n+2)$-х переменных.

Если дифференциальное уравнение n-го порядка можно записать в виде

$$y^{(n)} = f(x, y, y', ..., y^{(n-1)}),$$

то такое уравнение называется уравнением *в нормальной форме* или уравнением, *разрешенным относительно старшей производной*.

Решением дифференциального уравнения n-го порядка на интервале (a, b) называется n раз непрерывно дифференцируемая функция $y = \varphi(x)$, которая обращает данное уравнение в тождество на этом интервале.

Задача Коши (или *начальная задача*) для обыкновенного дифференциального уравнения n-го порядка состоит в следующем: найти решение уравнения

$$F(x, y, y', ..., y^{(n)}) = 0,$$

удовлетворяющее *начальным условиям*

$$y(x_0) = y_0, \quad y'(x_0) = y_1, \; \dots, \; y^{(n-1)}(x_0) = y_{n-1},$$

где x_0, y_0, y_1, y_{n-1} – заданные числа.

Рассмотрим задачу Коши для ОДУ первого порядка:

$$\begin{cases} y' = f(x, y), \\ y(x_0) = y_0. \end{cases}$$

Будем считать, что функция $f(x, y)$ в некоторой области удовлетворяет всем необходимым требованиям и задача поставлена корректно, т.е. решение задачи Коши существует и единственно. В большинстве случаев интегрирование таких уравнений невозможно не только в элементарных, но и в специальных функциях. Поэтому были созданы различные методы приближенного решения дифференциальных уравнений – аналитические, графические, численные. В результате применения численных методов искомую функцию получают ***в табличном виде***, т.е. в виде таблиц значений функции $y(x)$ в узловых точках x_i.

Выберем достаточно малый шаг h и построим систему равноотстоящих точек (сетку)

$$x_i = x_0 + ih, \ \ i = 0, 1, 2, \ldots,$$

которые называются узлами сетки.

Выполним дискретизацию задачи Коши, т.е. заменим дифференциальное уравнение разностным уравнением

$$y_{i+1} = \Phi(x_i, y_{i-p+1}, y_{i-p+2}, \ldots, y_i, y_{i+1}) \qquad (i \geq p-1),$$

которое необходимо решить на каждом шаге для нахождения y_{i+1}.

Выбор функции Ф определяет метод численного решения дифференциального уравнения. Если она не зависит от y_{i+1}, то получают ***явный*** метод (явную формулу для вычисления y_{i+1}), в противном случае – ***неявный*** метод. Метод, дающий формулу для вычисления y_{i+1} по m предыдущим значениям y_{i-m+1}, y_{i-m+2}, …, y_i, называется m-шаговым. Существуют две группы численных методов решения задачи Коши: одношаговые (или методы Рунге--Кутта) и многошаговые разностные методы.

Говорят, что ***метод сходится*** *в точке* x^*, если построена последовательность сеток, таких что

$$x^* = x_0 + nh \quad (h \to 0, n \to \infty) \ \text{ и } \ y(x^*) - y_n \to 0 \ \text{ при } \ h \to 0.$$

Если существует такое $p > 0$, что $y(x^*) - y_n = O(h^p)$ при $h \to 0$, то говорят, что метод имеет ***p-й порядок точности***.

Погрешность метода численного решения в точке x_i определяется нормой разности $y(x_i) - y_i$, где $y(x_i)$ – точное, а y_i – приближенное решение. ***Локальной погрешностью*** называют ошибку на данном шаге при условии, что предыдущие значения верны. ***Глобальная погрешность*** – это разность между вычисленным и точным решением, определяемым начальным условием.

Метод Эйлера (метод ломаных).

Этот метод является простейшим явным одношаговым методом.

Найдем решение уравнения $y' = f(x, y)$, удовлетворяющее начальному условию $y(x_0) = y_0$.

Согласно методу Эйлера уравнение $y' = f(x, y)$ заменяется разностным уравнением

$$\frac{y_{i+1} - y_i}{h} = f(x_i, y_i),$$

где $x_i = x_0 + ih$, $i = 0, 1, 2, \ldots$ Решение этого уравнения находится явным образом по рекуррентной формуле

$$y_{i+1} = y_i + hf(x_i, y_i), \ \ i = 0, 1, 2, \ldots$$

Геометрически, метод Эйлера состоит в том, что интегральную кривую $y = y(x)$, проходящую через точку (x_0, y_0) (т.е. график решения задачи Коши) заменяют ломаной с вершинами в точках $M_i(x_i, y_i)$. Эта ломаная называется ***ломаной Эйлера***. Ее звенья $M_i M_{i+1}$ – отрезки прямых с угловыми коэффициентами $f(x_i, y_i)$.

На каждом шаге метод Эйлера дает ***погрешность*** порядка h^2 (***локальная*** погрешность).

Глобальная погрешность метода имеет первый порядок, т.е. равна $O(h)$.

Метод Эйлера-Коши.

Этот метод является модификацией метода Эйлера.

Сначала по формуле метода Эйлера находят «грубое» приближение (или *предиктор*)

$$y_{i+1}^{(0)} = y_i + hf(x_i, y_i),$$

которое затем уточняют по формуле

$$y_{i+1} = y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_{i+1}, y_{i+1}^{(0)})\right),$$

получая так называемый *корректор*.

Локальная погрешность метода – $O(h^3)$, ***глобальная*** погрешность – $O(h^2)$.

Метод Рунге-Кутта.

Идея построения методов Рунге-Кутта порядка p состоит в следующем. Приближения к решениям $y(x_{i+1})$ ищут по формуле вида

$$y_{i+1} = y_i + h\varphi(x_i, y_i, h),$$

где $\varphi(x, y, h)$ – некоторая функция, приближающая отрезок ряда Тейлора p-го порядка и не содержащая частных производных функции $f(x, y)$:

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{1}{2}h^2 y''(x_i) + \ldots + \frac{1}{p!}h^p y^{(p)}(x_i) + o(h^{p+1}).$$

Метод Эйлера является методом Рунге-Кутта первого порядка точности. Для построения методов Рунге-Кутта порядка $p > 1$ функцию $\varphi(x, y, h)$ полагают зависящей от нескольких параметров. Их значения подбирают, сравнивая выражение $y_{i+1} = y_i + h\varphi(x_i, y_i, h)$ с многочленом Тейлора степени p.

Расчетные формулы ***метода Рунге-Кутта четвертого порядка*** имеют вид:

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)} + k_4^{(i)}\right),$$

где

$$k_1^{(i)} = hf(x_i, y_i), \quad k_2^{(i)} = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1^{(i)}}{2}\right),$$

$$k_3^{(i)} = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2^{(i)}}{2}\right), \quad k_4^{(i)} = hf\left(x_i + h, y_i + k_3^{(i)}\right),$$

причем $x_i = x_0 + ih$, $i = 0, 1, 2, \ldots$

Локальная погрешность метода – $O(h^5)$, ***глобальная*** погрешность – $O(h^4)$.

Формулы более высокого порядка точности, как правило, не используются в связи с их громоздкостью, возрастающей значительно быстрее, чем точность формулы.

Численное решение задачи Коши для системы обыкновенных дифференциальных уравнений.

Метод Эйлера.

Формула Эйлера легко обобщается на случай систем обыкновенных дифференциальных уравнений в нормальной форме:

$$\begin{cases} y_1' = f_1(x, y_1, \ldots, y_n), \\ y_2' = f_2(x, y_1, \ldots, y_n), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y_n' = f_n(x, y_1, \ldots, y_n) \end{cases}$$

с начальными условиями

$$y_1(x_0) = y_1^{(0)}, \quad y_2(x_0) = y_2^{(0)}, \ \ldots, \ y_n(x_0) = y_n^{(0)}.$$

Приближенные значения $y_k^{(i)}$ точного решения $y_k(x_i)$ вычисляются по формулам:

$$y_k^{(i)} = y_k^{(i-1)} + hf_k(x_{i-1}, y_1^{(i-1)}, y_2^{(i-1)}, \ldots, y_n^{(i-1)}), \quad k = \overline{1, n}, \quad i = 0, 1, 2, \ldots$$

В частности, для системы двух уравнений вида

$$\begin{cases} y' = f(x, y, z), \\ z' = g(x, y, z) \end{cases}$$

с начальными условиями

$$y(x_0) = y_0, \quad z(x_0) = z_0$$

расчетные формулы метода Эйлера имеют вид

$$y_{i+1} = y_i + h\, f(x_i, y_i, z_i), \quad z_{i+1} = z_i + h\, g(x_i, y_i, z_i),$$

где $x_i = x_0 + ih$, $i = 0, 1, 2, \ldots$

Метод Рунге-Кутта.

Расчетные формулы метода Рунге-Кутта четвертого порядка для системы двух уравнений

$$\begin{cases} y' = f(x, y, z), \\ z' = g(x, y, z) \end{cases}$$

с начальными условиями

$$y(x_0) = y_0, \quad z(x_0) = z_0$$

имеют вид

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)} + k_4^{(i)}\right),$$

$$z_{i+1} = z_i + \frac{1}{6}\left(r_1^{(i)} + 2r_2^{(i)} + 2r_3^{(i)} + r_4^{(i)}\right)$$

где

$$k_1^{(i)} = hf(x_i, y_i, z_i), \quad r_1^{(i)} = hg(x_i, y_i, z_i)$$

$$k_2^{(i)} = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1^{(i)}}{2}, z_i + \frac{r_1^{(i)}}{2}\right), \quad r_2^{(i)} = hg\left(x_i + \frac{h}{2}, y_i + \frac{k_1^{(i)}}{2}, z_i + \frac{r_1^{(i)}}{2}\right),$$

$$k_3^{(i)} = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2^{(i)}}{2}, z_i + \frac{r_2^{(i)}}{2}\right), \quad r_3^{(i)} = hg\left(x_i + \frac{h}{2}, y_i + \frac{k_2^{(i)}}{2}, z_i + \frac{r_2^{(i)}}{2}\right),$$

$$k_4^{(i)} = hf\left(x_i + h, y_i + k_3^{(i)}, z_i + r_3^{(i)}\right), \quad r_4^{(i)} = hg\left(x_i + h, y_i + k_3^{(i)}, z_i + r_3^{(i)}\right),$$

причем $x_i = x_0 + ih$, $i = 0, 1, 2, \ldots$

Основные функции пакета Mathematica, используемые при решении дифференциальных уравнений и их систем.

***DSolve*[*eqn*, *y*[*x*], *x*]** – решает дифференциальное уравнение *eqn* с функцией *y* и независимой переменной *x*.

***DSolve*[*eqn*, *y*[*x*], $\{x, a, b\}$] –** находит решение дифференциального уравнения *eqn* относительно функции $y(x)$ на отрезке $[a, b]$.

***DSolve*[{*eqn*1, *eqn*2, …}, {*y*1[*x*], *y*2[*x*], …}, *x*]** – находит решение системы дифференциальных уравнений *eqn*1, *eqn*2, … относительно функций *y*1, *y*2, … с независимой переменной *x*.

***NDSolve*[*eqn*, *y*[*x*], $\{x, a, b\}$]** – находит численное решение дифференциального уравнения *eqn* относительно функции $y(x)$ на отрезке $[a, b]$.

***NDSolve*[{*eqn*1, *eqn*2, …}, {*y*1[*x*], *y*2[*x*], …}, $\{x, a, b\}$]** – находит численное решение системы дифференциальных уравнений *eqn*1, *eqn*2, … относительно функций *y*1, *y*2, … на отрезке $[a, b]$.

***Floor*[*x*]** – выделяет целую часть выражения *x*;

***Floor*[*x*, *a*]** – определяет набольшее число, кратное *a*, которое меньше или равно *x*.

***Prepend*[*expr*, *elem*] –** добавляет в начало списка *expr* список значений *elem*.

***ListPlot*[*list*, *PlotJoined*→*True*]** – строит точки из списка *list* и соединяет их отрезками прямых.

Примеры численного решения дифференциальных уравнений и их систем средствами пакета Mathematica.

Пример 1. Решить задачу Коши $y' = -xy$, $y(0) = 1$:

а) методом Эйлера на отрезке $[0, 1]$ с шагом $h = 0{,}1$;

б) с помощью функции **DSolve**.

Сравнить полученные решения в узлах таблицы. Проиллюстрировать графиками.

Δ а) Введем функцию $f(x, y)$ – правую часть уравнения, границы отрезка, начальные значения x_0, y_0, шаг *h*, найдем число точек разбиения отрезка *n* (не считая x_0):

```
In[1]:= f[x_, y_] = -x y;

In[2]:= a = 0; b = 1; x0 = 0; y0 = 1; h = 0.1; n = Floor[(b - a)/h];
                                                 округление вниз
```

Составим таблицу приближенных значений функции $y(x)$, вычисленных с помощью метода Эйлера:

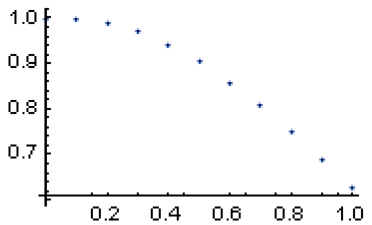
```
In[3]:= x = x0; y = y0; eul1 = Table[{x, y} = {x + h, y + h f[x, y]}, {i, n}];
                               таблица значений

In[4]:= eul1 = Prepend[eul1, {x0, y0}]
               добавить в начало

Out[4]= {{0, 1}, {0.1, 1.}, {0.2, 0.99}, {0.3, 0.9702},
  {0.4, 0.941094}, {0.5, 0.90345}, {0.6, 0.858278},
  {0.7, 0.806781}, {0.8, 0.750306}, {0.9, 0.690282}, {1., 0.628157}}
```

Изобразим полученные точки на графике:

```
In[5]:= gr1 = ListPlot[eul1, ImageSize → Small]
              диаграмма разб⋯  размер изо⋯  малый
```

Out[5]=


б) Решим задачу Коши с помощью функции **DSolve**.

```
In[6]:= Clear[x, y]
        очистить

In[7]:= sol = DSolve[{y'[x] == f[x, y[x]], y[x0] == y0}, y[x], x]
              решить дифференциальные уравнения
```

Out[7]= $\left\{\left\{y[x] \to e^{-\frac{x^2}{2}}\right\}\right\}$

Решение сохраним в виде функции $y1(x)$:

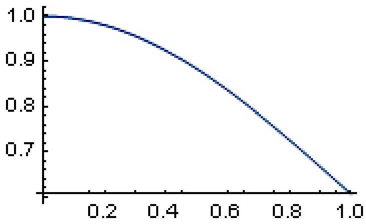
```
In[8]:= y1[x_] = y[x] /. Flatten[sol]
                         уплостить
```

Out[8]= $e^{-\frac{x^2}{2}}$

Построим ее график на отрезке $[0,1]$:

```
In[9]:= gr2 = Plot[y1[x], {x, 0, 1}, ImageSize → Small]
              |график функции            |размер изо⋯ |малый
```

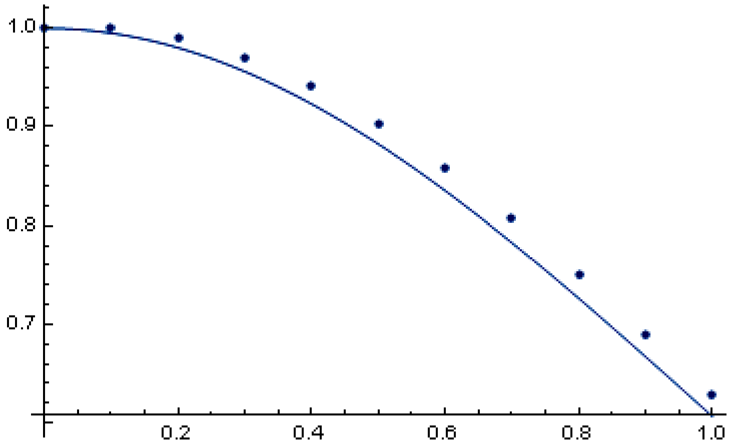
Out[9]=



Сравним полученные решения графически:

```
In[10]:= Show[gr1, gr2, ImageSize → Medium]
         |показать        |размер изо⋯ |средний
```

Out[10]=



Оценим погрешность приближенного решения, найдя вектор ошибки и его норму:

```
In[11]:= delta = Table[y1[x0 + (i - 1) h] - eul1[[i, 2]], {i, n + 1}]
                 |таблица значений

Out[11]= {0., -0.00498752, -0.00980133, -0.0142025, -0.0179777, -0.0209533,
          -0.0230075, -0.0240765, -0.0241574, -0.0233051, -0.0216258}

In[12]:= Norm[delta]
         |норма

Out[12]= 0.0615491
```

▲

Пример 2. Решить задачу Коши $y' = 0{,}3x + y^2$, $y(0) = 0{,}6$:

а) методом Рунге-Кутта 4-го порядка на отрезке $[0,1]$ с шагом $h = 0{,}1$;

б) с помощью функций **DSolve** и **NDSolve**.

Сравнить полученные решения.

Δ а) Введем функцию $f(x, y)$ – правую часть уравнения, границы отрезка, начальные значения x_0, y_0, шаг h, найдем число точек разбиения отрезка n (не считая x_0):

```
In[1]:= f[x_, y_] = 0.3 x + y^2;

In[2]:= a = 0; b = 1; x0 = 0; y0 = 0.6; h = 0.1; n = Floor[(b - a)/h];
                                                    округление вниз
```

Организуем цикл и создадим таблицу **sol1** приближенных значений решения дифференциального уравнения, полученных с помощью метода Рунге-Кутта:

```
In[3]:= sol1 = List[{x0, y0}];
               список

In[4]:= x = x0; y = y0;
        For[k = 1, k < n + 1, k++,
        цикл ДЛЯ
          k1[x_, y_] := h*f[x, y];
          k2[x_, y_] := h*f[x + h/2, y + k1[x, y]/2];
          k3[x_, y_] := h*f[x + h/2, y + k2[x, y]/2];
          k4[x_, y_] := h*f[x + h, y + k3[x, y]];
          x = x + h; y = y + (k1[x, y] + 2*k2[x, y] + 2*k3[x, y] + k4[x, y])/6;
          sol1 = Append[sol1, {x, y}]]
                 добавить в конец
```

Выведем таблицу:

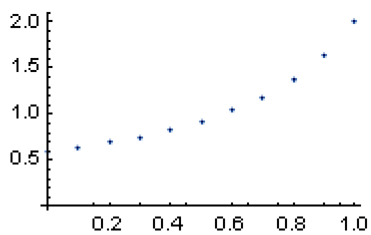
```
In[6]:= sol1

Out[6]= {{0, 0.6}, {0.1, 0.643059}, {0.2, 0.695247},
         {0.3, 0.758465}, {0.4, 0.835315}, {0.5, 0.929462}, {0.6, 1.04623},
         {0.7, 1.19367}, {0.8, 1.38454}, {0.9, 1.6403}, {1., 2.00017}}
```

Изобразим полученные точки на графике:

```
In[7]:= gr1 = ListPlot[sol1, ImageSize → Small]
              диаграмма разб⋯  размер изо⋯  малый
```

Out[7]=



б) Решим данную задачу Коши с помощью функций **DSolve** и **NDSolve**. Полученные аналитическое и численное решения обозначим **sol2** и **sol3** соответственно.

```
In[8]:= Clear[x, y]
        очистить

In[9]:= sol2 = DSolve[{y'[x] == f[x, y[x]], y[x0] == y0}, y[x], x];
               решить дифференциальные уравнения

        y1[x_] = y[x] /. Flatten[sol2]
                         уплостить
```

Out[10]=

$$-\left(\left(0.273861\left(1.\,x^{3/2}\,\mathrm{BesselJ}\left[-\frac{4}{3},\,0.365148\,x^{3/2}\right]-2.4589\,x^{3/2}\,\mathrm{BesselJ}\left[-\frac{2}{3},\,0.365148\,x^{3/2}\right]+1.82574\,\mathrm{BesselJ}\left[-\frac{1}{3},\,0.365148\,x^{3/2}\right]-1.\,x^{3/2}\,\mathrm{BesselJ}\left[\frac{2}{3},\,0.365148\,x^{3/2}\right]\right)\right)\Big/\left(x\left(1.\,\mathrm{BesselJ}\left[-\frac{1}{3},\,0.365148\,x^{3/2}\right]-1.22945\,\mathrm{BesselJ}\left[\frac{1}{3},\,0.365148\,x^{3/2}\right]\right)\right)\right)$$

Как видно, аналитическое решение выражается через функции Бесселя.

Численное решение система **Mathematica** выдает в виде некоторой интерполирующей функции:

```
In[11]:= sol3 = NDSolve[{y'[x] == f[x, y[x]], y[x0] == y0}, y[x], {x, 0, 1}]
                численно решить ДУ

Out[11]= {{y[x] → InterpolatingFunction[ Domain: {{0., 1.}}
                                          Output: scalar ][x]}}
```

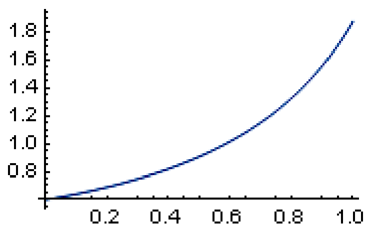
Заметим, что по умолчанию функция **NDSolve** использует метод Рунге-Кутта.

Построим графики этих решений.

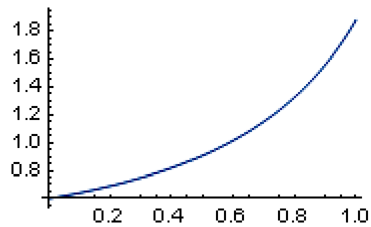
```
In[12]:= gr2 = Plot[y1[x], {x, 0, 1}, ImageSize → Small]
               график функции      размер изо… малый

In[13]:= gr3 = Plot[Evaluate[y[x] /. sol3], {x, 0, 1}, ImageSize → Small]
               гра… вычислить                        размер изо… малый
```

Out[12]=



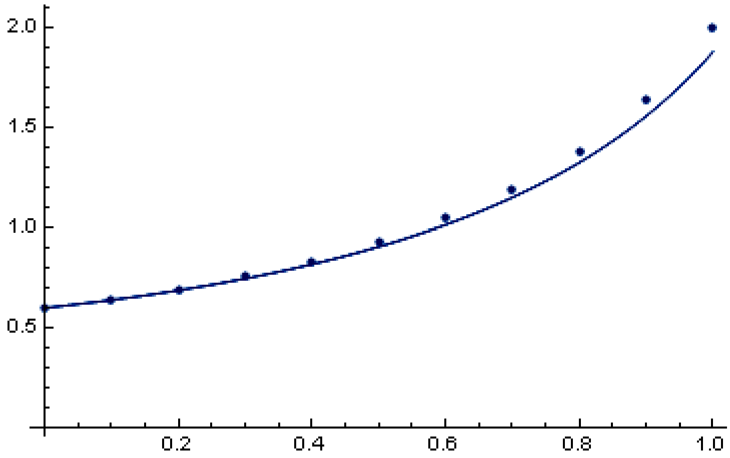
Out[13]=



Сравним решения, полученные тремя способами (методом Рунге-Кутта, с помощью функций **DSolve** и **NDSolve**), графически:

In[14]:= `Show[gr1, gr2, gr3, ImageSize → Medium]`
показать размер изо⋯ средний

Out[14]=



Графики решений **sol2** и **sol3** совпали.

Очевидно, метод Рунге-Кутта дает более точное приближенное решение, чем метод Эйлера. ▲