

Пакеты научных вычислений

Лекция 7

Решение дифференциальных уравнений в Maple

Аналитические решения ОДУ. Приближенные решения ОДУ.
Численные решения ОДУ. Графическое представление
решений, построение фазовых портретов.

Решение уравнений в частных производных.

Обзор пакетов Maple

Наседкина А. А.



Решение обыкновенных дифференциальных уравнений в Maple: аналитическое решение

- Команды `dsolve` и `odetest`
- Аналитические решения ОДУ
 - Общее решение
 - Фундаментальная система решений
 - Решение задачи Коши или краевой задачи
 - Графическое представление решения
 - Решение систем дифференциальных уравнений
 - Интегральные преобразования для решения дифференциальных уравнений

Команды dsolve и odetest

Решение ОДУ (обыкновенных дифференциальных уравнений)

- **dsolve(ODE)**
- **dsolve(ODE, y(x), options)**
- **dsolve({ODE, ICs}, y(x), options)**

ODE – ОДУ либо система ОДУ в виде множества или списка

y(x) – неопределенная функция одной переменной либо множество или список таких функций (неизвестные уравнения)

ICs – начальные условия в виде $y(x_0)=a$, $D(y)(x_0)=b$, ..., $(D@@n)(y)(x_0)=c$

options – дополнительные опции, например:

- **type=exact** (аналитическое решение – опция по умолчанию),
- **type=series** (приближенное решение в виде степенного ряда) ,
- **type=numeric** (численное решение),
- **output=basis** (фундаментальная система решений)
- **method=laplace** (решение с помощью интегральных преобразований)

Проверка найденного решения

- **odetest(sol,ode)**

Аналитическое решение ОДУ

- **dsolve(ode, var, options)**
- Аналитическое решение ищется по умолчанию(**type=exact**).
- При невозможности выделить искомую функцию решение выводится в неявном виде. Для вывода решения в явном виде следует указать опцию **type=explicit** либо задать **_EnvExplicit:=true** перед выполнением команды **dsolve**.
- Общее решение дифференциального уравнения зависит от произвольных постоянных, число которых равно порядку дифференциального уравнения. В *Maple* такие постоянные, как правило, обозначаются как *_C1*, *_C2*, и т.д.
- В строке вывода решение неоднородного линейного ОДУ всегда состоит из слагаемых, которые содержат произвольные постоянные (это *общее решение* соответствующего **однородного** ОДУ), и слагаемых без произвольных постоянных (это *частное решение* этого же **неоднородного** ОДУ).
- Команда **dsolve** выдает решение дифференциального уравнения в формате вида $y(x)=\dots$. Для того, чтобы с решением можно было бы работать далее (например, построить график решения) следует отделить правую часть полученного решения командой **rhs(%)**.

Аналитическое решение одного ОДУ: примеры

ОДУ первого порядка

```
> restart;  
de := diff(y(x), x) + y(x) * cos(x) = sin(x) * cos(x);  
dsolve(de, y(x));
```

$$de := \frac{d}{dx} y(x) + y(x) \cos(x) = \sin(x) \cos(x)$$
$$y(x) = \sin(x) - 1 + e^{-\sin(x)} _C1 \quad (1)$$

Использование нотации со штрихами

```
> dsolve(y' + y * cos(x) = sin(x) * cos(x));
```

$$y(x) = \sin(x) - 1 + e^{-\sin(x)} _C1 \quad (2)$$

ОДУ второго порядка

```
> restart; deq := \frac{d^2}{dx^2} y(x) - 2 \left( \frac{d}{dx} y(x) \right) + y(x) = \sin(x) + e^{-x} :
```

```
> dsolve(deq, y(x));
```

$$y(x) = e^x _C2 + e^x x _C1 + \frac{1}{4} e^{-x} (2 \cos(x) e^x + 1) \quad (3)$$

Использование нотации со штрихами

```
> dsolve(y'' - 2 * y' + y = sin(x) + e^{-x});  
> restart;
```

$$y(x) = e^x _C2 + e^x x _C1 + \frac{1}{4} e^{-x} (2 \cos(x) e^x + 1) \quad (4)$$

Фундаментальная система решений ОДУ

- **dsolve(ode, var, output=basis)** – вывод фундаментальной системы решений (базисных функций) ОДУ

Общее решение (нотация со штрихами)

$$\begin{aligned} &> \text{dsolve}(y'''' + 2 \cdot y'' + y = 0, y); \\ &\quad y(x) = _C1 \sin(x) + _C2 \cos(x) + _C3 \sin(x) x + _C4 \cos(x) x \end{aligned} \quad (5)$$

Фундаментальная система решений

$$\begin{aligned} &> \text{dsolve}(y'''' + 2 \cdot y'' + y = 0, y, \text{output} = \text{basis}); \\ &\quad [\sin(x), \cos(x), \cos(x) x, \sin(x) x] \end{aligned} \quad (6)$$

Стандартная запись

$$\begin{aligned} &> de := \text{diff}(y(x), x\$4) + 2 * \text{diff}(y(x), x\$2) + y(x) = 0; \\ &\quad de := \frac{d^4}{dx^4} y(x) + 2 \left(\frac{d^2}{dx^2} y(x) \right) + y(x) = 0 \end{aligned} \quad (7)$$

$$\begin{aligned} &> \text{dsolve}(de, y(x), \text{output} = \text{basis}); \\ &\quad [\sin(x), \cos(x), \cos(x) x, \sin(x) x] \end{aligned} \quad (8)$$

Решение задачи Коши или краевой задачи

- Команда **dsolve** может найти решение задачи Коши или краевой задачи, если помимо ОДУ задать начальные или краевые условия для неизвестной функции.

- dsolve({ODE, ICs}, y(x), options)**

Запись начальных или краевых условий

- с помощью дифференциального оператора

$y(a)=ya, D(y)(b)=yb, (D@@2)(y)(c)=yc, (D@@3)(y)(d)=yd,..., (D@@n)(y)(e)=ye$

- с помощью нотации со штрихами (2D-Math):

$$y(a) = ya, y'(b) = yb, y''(c) = yc, y'''(d) = yd, ..., y^{(n)}(e) = ye$$

Первый способ

> $de := diff(y(x), x\$4) + diff(y(x), x\$2) = 2 * \cos(x);$

$$de := \frac{d^4}{dx^4} y(x) + \frac{d^2}{dx^2} y(x) = 2 \cos(x)$$

> $cond := y(0) = -2, D(y)(0) = 1, (D@@2)(y)(0) = 0, (D@@3)(y)(0) = 0;$

$$cond := y(0) = -2, D(y)(0) = 1, D^{(2)}(y)(0) = 0, D^{(3)}(y)(0) = 0$$

> $dsolve(\{de, cond\}, y(x));$

$$y(x) = -2 \cos(x) - \sin(x) x + x$$

Второй способ

> $dsys := \{y'''' + y'' = 2 \cdot \cos(x), y(0) = -2, y'(0) = 1, y''(0) = 0, y'''(0) = 0\};$

> $dsolve(dsys, y(x));$

$$y(x) = -2 \cos(x) - \sin(x) x + x$$

Решение краевой задачи: точное решение и его график

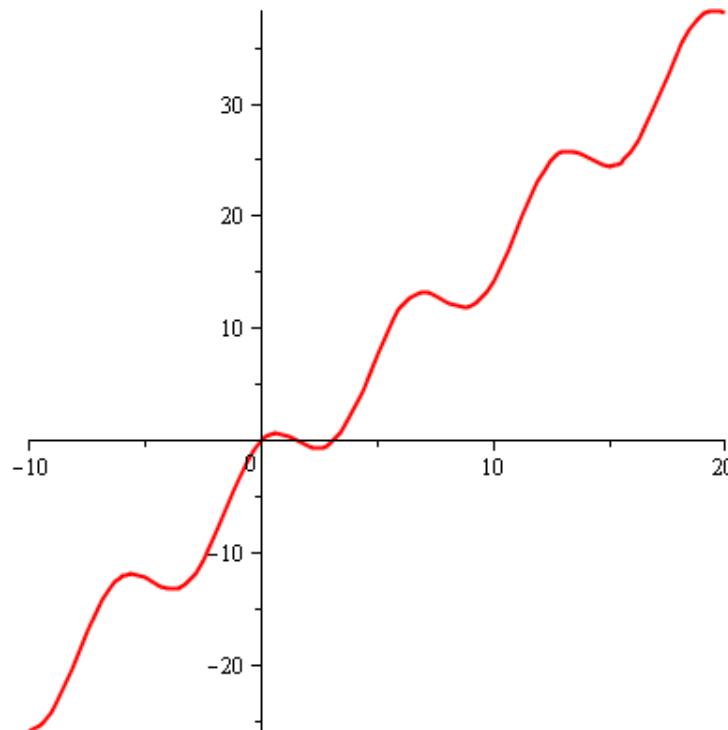
```
> restart;  
> de := diff(y(x), x$2) + y(x) = 2*x - pi; cond := y(0) = 0, y(pi/2) = 0;  
dsolve({de, cond}, y(x));
```

$$de := \frac{d^2}{dx^2} y(x) + y(x) = 2x - \pi$$

$$cond := y(0) = 0, y\left(\frac{1}{2}\pi\right) = 0$$

$$y(x) = \cos(x) \pi + 2x - \pi$$

```
> y1 := rhs(%): plot(y1, x=-10..20, thickness=2);
```



Решение систем ОДУ

- **dsolve({sys},{x(t), y(t),...}, options)** – решение системы ОДУ
- **dsolve({sys,cond}, {x(t),y(t),...}, options)** – решение задачи Коши или краевой задачи с системой ОДУ; **x(t), y(t),...** – набор неизвестных функций

```
> restart; sys:=diff(x(t),t)=-4*x(t)-2*y(t)+2/(exp(t)-1),
diff(y(t),t)=6*x(t)+3*y(t)-3/(exp(t)-1);
```

$$\text{sys} := \frac{d}{dt} x(t) = -4x(t) - 2y(t) + \frac{2}{e^t - 1}, \frac{d}{dt} y(t) = 6x(t) + 3y(t) - \frac{3}{e^t - 1}$$

Список и множество

```
> sol1:=dsolve([sys], [x(t), y(t)]);
```

$$\text{sol1} := \left\{ x(t) = -\frac{1}{2} _C2 + 2e^{-t} + 2e^{-t} \ln(e^t - 1) + \frac{2}{3} e^{-t} _C1, y(t) = \frac{-3 - 3 \ln(e^t - 1) - _C1}{e^t} + _C2 \right\}$$

```
> sol2:=dsolve({sys}, {x(t), y(t)});
```

$$\text{sol2} := \left\{ x(t) = \frac{2 + 2 \ln(e^t - 1) - _C1}{e^t} + _C2, y(t) = -2 _C2 - 3e^{-t} - 3e^{-t} \ln(e^t - 1) + \frac{3}{2} e^{-t} _C1 \right\}$$

```
> simplify(rhs(sol1[1]));simplify(rhs(sol2[1]));
```

$$-\frac{1}{2} _C2 + 2e^{-t} + 2e^{-t} \ln(e^t - 1) + \frac{2}{3} e^{-t} _C1$$

$$2e^{-t} + 2e^{-t} \ln(e^t - 1) - e^{-t} _C1 + _C2$$

```
> simplify(rhs(sol1[2]));simplify(rhs(sol2[2]));
```

$$-3e^{-t} - 3e^{-t} \ln(e^t - 1) - e^{-t} _C1 + _C2$$

$$-2 _C2 - 3e^{-t} - 3e^{-t} \ln(e^t - 1) + \frac{3}{2} e^{-t} _C1$$

Проверка найденного решения

```
> odetest(sol1,[sys]);
```

$$[0, 0]$$

```
> odetest(sol2,{sys});
```

$$\{0\}$$

Решение системы ОДУ с начальными условиями (задача Коши): примеры

```
> restart,  
> sys := diff(x(t), t, t) + 5·diff(x(t), t) + 2·diff(y(t), t) + y(t) = 0, 3·diff(x(t), t, t) + 5·x(t)  
+ diff(y(t), t) + 3·y(t) = 0, x(0) = 1, (D(x))(0) = 0, y(0) = 1;
```

```
sys :=  $\frac{d^2}{dt^2} x(t) + 5 \left( \frac{d}{dt} x(t) \right) + 2 \left( \frac{d}{dt} y(t) \right) + y(t) = 0, 3 \left( \frac{d^2}{dt^2} x(t) \right) + 5 x(t) + \frac{d}{dt} y(t)$   
+ 3 y(t) = 0, x(0) = 1, D(x)(0) = 0, y(0) = 1
```

```
> sol := dsolve( {sys}, {x(t), y(t)} )  
sol :=  $\left\{ x(t) = -\frac{1}{2} e^{-t} + \frac{3}{2} e^t - 2 e^t t, y(t) = 2 e^{-t} - e^t + 4 e^t t \right\}$ 
```

Проверка найденного решения

```
> odetest(sol, {sys});
```

{0}

Решение ОДУ с помощью интегральных преобразований

- Для решения дифференциальных уравнений и задач с начальными данными можно применять интегральные преобразования.
- Для этого в качестве опций команды **dsolve** нужно задать метод интегральных преобразований в **method=transform**, где в качестве **transform** можно указать: **laplace**, **fourier**, **fouriercos**, **fouriersin**.

```
> ode := diff(y(x), x, x) = y(x)+1; icond:=y(0) = 1, (D(y))(0) = 0;
```

$$ode := \frac{d^2}{dx^2} y(x) = y(x) + 1$$

$$icond := y(0) = 1, D(y)(0) = 0$$

```
> dsolve({ode,icond}, y(x));
```

$$y(x) = e^{-x} + e^x - 1$$

```
> dsolve({ode,icond}, y(x), method=laplace);
```

$$y(x) = -1 + 2 \cosh(x)$$

Решение обыкновенных дифференциальных уравнений в Maple: приближенное и численное решение

- Приближенные решения
- Численные решения
- График численного решения, команда `odeplot`

Приближенные решения ОДУ

- Для многих типов ОДУ нельзя найти точное аналитическое решение. В этом случае ОДУ можно решить с помощью приближенных методов, и, в частности, с помощью разложения в степенной ряд неизвестной функции.
- **`dsolve(ODE, y(x), 'type=series')`**
`dsolve( {ODE, ICs}, y(x), 'type=series')`
`dsolve( {sysODE, ICs}, {funcs}, 'type=series')`
- Вместо **`type=series`** можно указывать просто **`series`**
- Для того, чтобы указать порядок разложения **`n`**, следует перед командой **`dsolve`** задать **`Order:=n`**.
- Если ищется общее решение ОДУ в виде разложения в степенной ряд, то коэффициенты при степенях x найденного разложения будут содержать неизвестные значения функции в нуле **`y(0)`** и ее производных **`D(y)(0)`**, **`(D@@2)(y)(0)`** и т.д.
- Для выделения *частного решения* следует задать начальные условия, количество которых должно совпадать с порядком соответствующего ОДУ.
- Разложение в степенной ряд имеет тип **`series`**. Для того, чтобы с решением можно было бы работать далее (например, построить график решения), его следует преобразовать в полином с помощью команды **`convert(%, polynom)`**, а затем выделить правую часть полученного выражения командой **`rhs(%)`**.

Приближенные решения ОДУ: примеры

Решение задачи Коши в виде степенного ряда с точностью до 5-го порядка.

```
> restart; Order:=5:  
> dsolve({diff(y(x),x)=y(x)+x*exp(y(x)), y(0)=0}, y(x), type=series);  
>
```

$$y(x) = \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{6}x^4 + O(x^5)$$

Общее решение ОДУ в виде разложения в степенной ряд до 4-го порядка

```
> restart; Order := 4; de := diff(y(x), x, x) - y(x)^3 = exp(-x)*cos(x);
```

$$de := \frac{d^2}{dx^2} y(x) - y(x)^3 = e^{-x} \cos(x)$$

```
> f:=dsolve(de,y(x),series);
```

$$f := y(x) = y(0) + D(y)(0)x + \left(\frac{1}{2}y(0)^3 + \frac{1}{2}\right)x^2 + \left(\frac{1}{2}y(0)^2 D(y)(0) - \frac{1}{6}\right)x^3 + O(x^4)$$

Решение при заданных начальных условиях

```
> subs(y(0)=1, D(y)(0)=0, f);
```

$$y(x) = 1 + x^2 - \frac{1}{6}x^3 + O(x^4)$$

Приближенные решения ОДУ: пример сравнения точного и приближенного решения

> restart, dsys := $y''' - y' = 3 \cdot (2 - x^2) \cdot \sin(x)$, $y(0) = 1$, $y'(0) = 1$, $y''(0) = 1$;

$dsys := \frac{d^3}{dx^3} y(x) - \left(\frac{d}{dx} y(x) \right) = 3 (2 - x^2) \sin(x)$, $y(0) = 1$, $D(y)(0) = 1$,

$D^{(2)}(y)(0) = 1$

Точное решение

> dsolve({dsys}, y(x));

$$y(x) = \frac{7}{4} e^x + \frac{3}{4} e^{-x} + \frac{21}{2} \cos(x) + 6 \sin(x) x - \frac{3}{2} \cos(x) x^2 - 12$$

Выделение правой части

> y1 := rhs(%);

$$y1 := \frac{7}{4} e^x + \frac{3}{4} e^{-x} + \frac{21}{2} \cos(x) + 6 \sin(x) x - \frac{3}{2} \cos(x) x^2 - 12$$

Приближенное решение

> Order := 6: dsolve({dsys}, y(x), series);

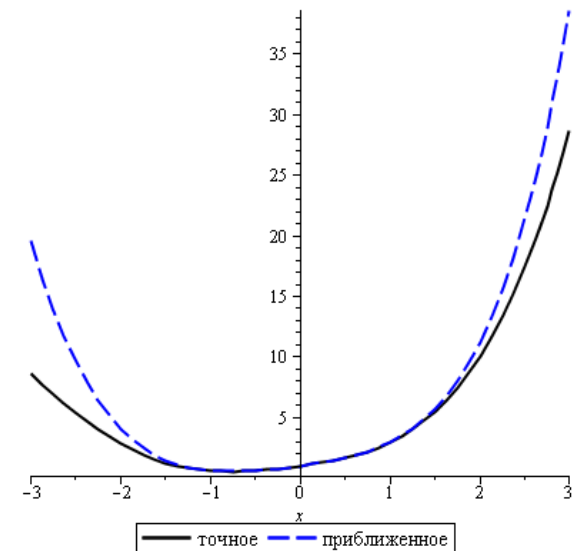
$$y(x) = 1 + x + \frac{1}{2} x^2 + \frac{1}{6} x^3 + \frac{7}{24} x^4 + \frac{1}{120} x^5 + O(x^6)$$

Конвертация в полином и выделение правой части

> y2:=convert(rhs(%),polynom);

$$y2 := 1 + x + \frac{1}{2} x^2 + \frac{1}{6} x^3 + \frac{7}{24} x^4 + \frac{1}{120} x^5$$

> plot([y1, y2], x = -3 .. 3, color = [black, blue], thickness = 2, linestyle = [solid, dash], legend=["точное", "приближенное"]);



Численное решение ОДУ и его график: команда odeplot

- **dsolve(ODE, y(x), 'type=numeric')**
dsolve({ODE, ICs}, y(x), 'type=numeric')
dsolve({sysODE, ICs}, {funcs}, 'type=numeric')
- Вместо **type=numeric** можно указывать просто **numeric**
- В дополнительных опциях можно указать метод численного интегрирования дифференциального уравнения в виде **method = numeric_method**, где в качестве **numeric_method** можно указать: **rkf45**, **rosenbrock**, **bvp**, **rkf45_dae**, **rosenbrock_dae**, **dverk78**, **lsode**, **gear**, **taylorseries**, **mebdfi**, **classical**.
- Например, **method=rkf45** – метод Рунге-Кутта-Фельберга 4-5-ого порядка (установлен по умолчанию); **method=dverk78** – метод Рунге-Кутта 7-8 порядка; **method=classical** – классический метод Рунге-Кутта 3-его порядка; **method=gear** – метод Гира.
- В результате выполнения команды dsolve с параметром **type=numeric** будет создана процедура, к которой можно обращаться для вычисления решения при заданном значении переменной и для построения графика на заданном промежутке.
- **plots[odeplot](sol_num, [x, y(x)], x=x1..x2, options)** – график численного решения ОДУ, где **sol_num** – численное решение, полученное с помощью команды **dsolve** с опцией **numeric**

Численные решения ОДУ: примеры

```
> restart; eq:=diff(y(x),x$2)-x*sin(y(x))=sin(2*x):  
> cond:=y(0)=0, D(y)(0)=1:  
> sol_num:=dsolve({eq,cond},y(x),numeric);  
      sol_num:=proc(x_rkf45) ... end proc
```

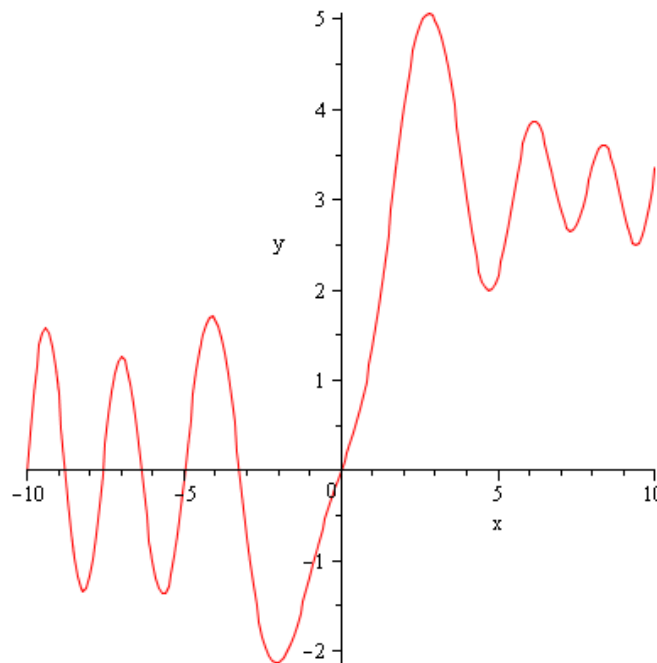
Значение решения при заданном значении переменной $x=0.5$

```
> sol_num(0.5);
```

$$\left[x = 0.5, y(x) = 0.544926322815993624, \frac{d}{dx} y(x) = 1.27250331302082364 \right]$$

График численного решения на интервале $(-10,10)$:

```
> with(plots): odeplot(sol_num,[x,y(x)],-10..10,numpoints=200);
```



Пример сравнения точного, численного и приближенного решения

```
> restart; eq:=diff(y(x), x, x) = y(x)+1; cond:=y(0)=0, D(y)(0)=1;
```

$$eq := \frac{d^2}{dx^2} y(x) = y(x) + 1$$

$$cond := y(0) = 0, D(y)(0) = 1$$

Точное решение

```
> sol_exact:=dsolve({eq,cond},y(x)); p_exact:=rhs(sol_exact):
```

$$sol_exact := y(x) = e^x - 1$$

Численное решение

```
> sol_num:=dsolve({eq,cond},y(x),numeric); p_num:=sol_num:
```

```
sol_num := proc(x_rkf45) ... end proc
```

Приближенное решение

```
> sol_series:=dsolve({eq,cond},y(x),series);
```

$$sol_series := y(x) = x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 + \frac{1}{120}x^5 + O(x^6)$$

```
> p_series:=rhs(convert(sol_series, polynom));
```

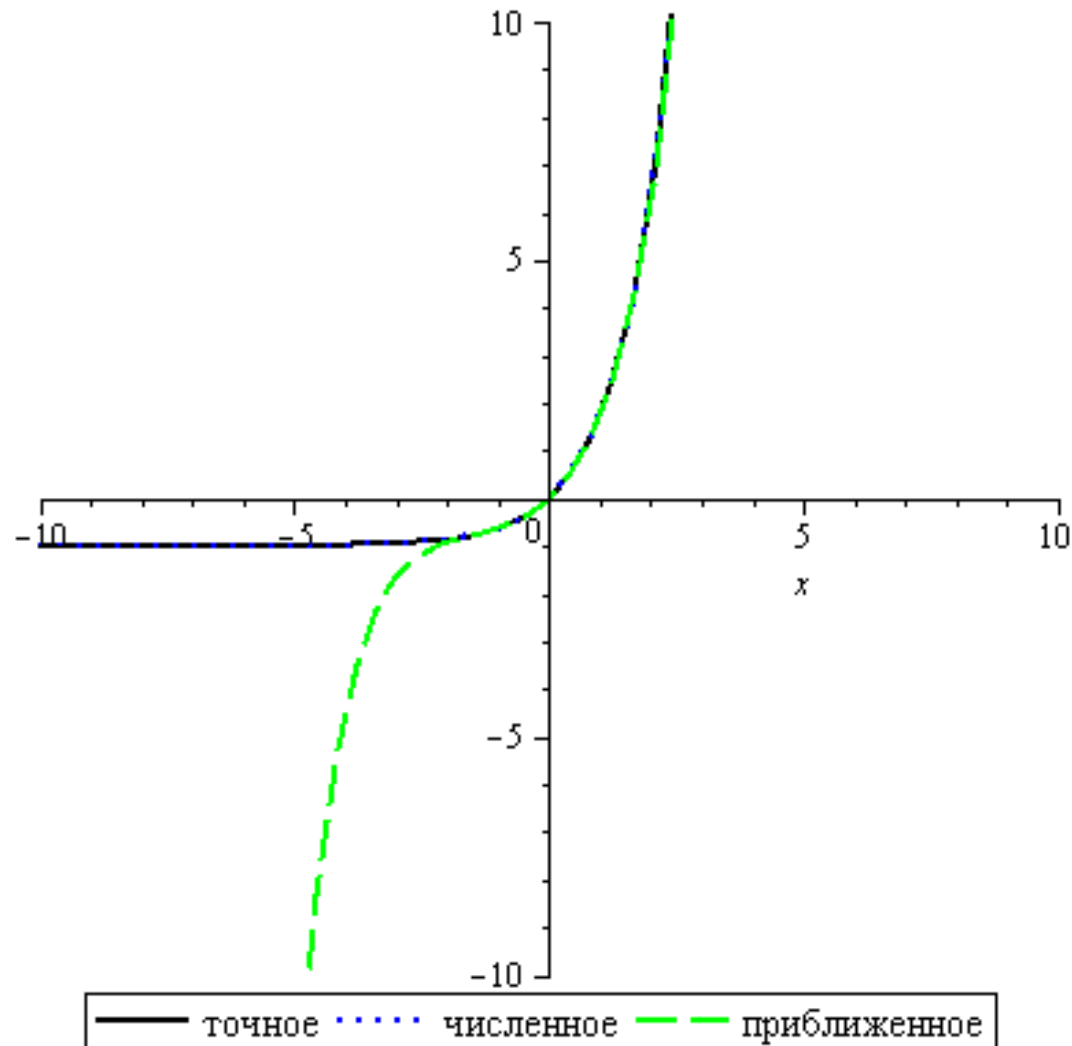
```
> with(plots): p1:=plot(p_exact,x=-10..10,-10..10, thickness=2, color=black,  
legend="точное");
```

```
> p2:=odeplot(p_num,[x,y(x)],x=-10..10,thickness=2, linestyle=2, color=blue,  
legend="численное");
```

```
> p3:=plot(p_series, x=-10..10, thickness=2, linestyle=3, color=green,  
legend="приближенное");
```

```
> display([p1,p2,p3]);
```

Пример сравнения точного, численного и приближенного решения: график



Объект DESol и пакет DEtools для работы с обыкновенными дифференциальными уравнениями

- Объект DESol для представления решения ОДУ
- Пакет DEtools: обзор команд
- Команда odeadvisor

Объект DESol

- **DESol** – объект (структура данных) для представления решения ОДУ; может использоваться, когда отсутствует явное представление решения
- **DESol(expr)**
DESol(expr, y)
- Объект DESol можно дифференцировать, интегрировать, находить разложение в ряд, находить численное решение и т. п.

$$\begin{aligned} &> de := DESol\left(\frac{d}{dx}y(x) - y(x), y(x)\right) \\ &\qquad\qquad\qquad de := DESol\left(\left\{\frac{d}{dx}y(x) - y(x)\right\}, \{y(x)\}\right) \\ &> \frac{\partial}{\partial x} de - de \end{aligned}$$

0

Пакет DEtools

- **DEtools** (синоним: **ODEtools**) – пакет команд для работы с ОДУ, содержит команды для исследования ОДУ, визуализации решений ОДУ, преобразования ОДУ и т. д.

> with(DEtools);

[AreSimilar, DEnormal, DEplot, DEplot3d, DEplot_polygon, DFactor, DFactorLCLM, DFactorsols, Dchangevar, FunctionDecomposition, GCRD, Gosper, Heunsols, Homomorphisms, IsHyperexponential, LCLM, MeijerGsols, MultiplicativeDecomposition, PDEchangecoords, PolynomialNormalForm, RationalCanonicalForm, ReduceHyperexp, RiemannPsols, Xchange, Xcommutator, Xgauge, Zeilberger, abelsol, adjoint, autonomous, bernoullisol, buildsol, buildsym, canoni, caseplot, casesplit, checkrank, chinisol, clairautsol, constcoeffsols, convertAlg, convertsys, dalembertsol, dcoeffs, de2diffop, dfieldplot, diff_table, diffop2de, dperiodic_sols, dpolyform, dsubs, eigenring, endomorphism_charpoly, equinv, ηk, eulersols, exactsol, expsols, exterior_power, firint, firtest, formal_sol, gen_exp, generate_ic, genhomosol, gensys, hamilton_eqs, hypergeomsols, hyperode, indicialeq, infgen, initialdata, integrate_sols, intfactor, invariants, kovaciccsols, leftdivision, liesol, line_int, linearsol, matrixDE, matrix_riccati, maxdimsystems, moser_reduce, muchange, mult, mutest, newton_polygon, normalG2, ode_int_y, ode_y1, odeadvisor, odepde, parametricsol, particularsol, phaseportrait, poincare, polysols, power_equivalent, ratsols, redode, reduceOrder, reduce_order, regular_parts, regularsp, remove_RootOf, riccati_system, riccatisol, rifread, rifsimp, rightdivision, rtaylor, separablesol, singularities, solve_group, super_reduce, symgen, symmetric_power, symmetric_product, symtest, transinv, translate, untranslate, varparam, zoom]

Команда odeadvisor из пакета DEtools

- `odeadvisor(ODE)`
- `odeadvisor(ODE, y(x), [type1, type2, ...], help)` – классификация ОДУ и предложение методов решения, запрос **help** дает вывод справочной страницы

```
> with(DEtools) : ODE1 := x  $\left( \frac{d}{dx} y(x) \right) + a y(x)^2 - y(x) + b x^2$ 
```

```
ODE1 := x  $\left( \frac{d}{dx} y(x) \right) + a y(x)^2 - y(x) + b x^2$ 
```

```
> odeadvisor(ODE1)
```

```
[[_homogeneous, class D], _rational, _Riccati]
```

```
> ODE2 := (2 y(x) - x)  $\left( \frac{d}{dx} y(x) \right) - y(x) - 2 x$ 
```

```
ODE2 := (2 y(x) - x)  $\left( \frac{d}{dx} y(x) \right) - y(x) - 2 x$ 
```

```
> odeadvisor(ODE2)
```

```
[[_homogeneous, class A], _exact, _rational, [_Abel, 2nd type, class A]]
```

Графическое представление решений обыкновенных дифференциальных уравнений

- Команда `odeplot` и графические команды пакета `DEtools`
- Двумерные графики решений, команда `DEplot`
- Трёхмерные графики решений, команда `DEplot3d`
- Построение фазового портрета и поля направлений
 - `DEplot`
 - `phaseportrait`
 - `dfieldplot`

Графические команды для представления решений ОДУ и систем ОДУ

Графические команды пакета **plots** для ОДУ

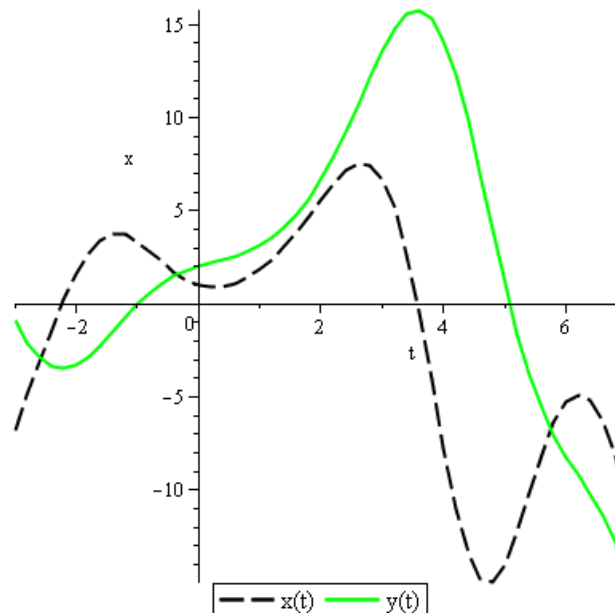
- **odeplot(dsn, vars, range, options)** – график численного решения ОДУ или системы ОДУ

Графические команды пакета **DEtools**

- **DEplot** – график численного решения ОДУ или системы ОДУ
- **DEplot3d** – 3D-график численного решения ОДУ или системы ОДУ
- **phaseportrait** – фазовый портрет для системы двух ОДУ первого порядка или одного ОДУ любого порядка; для *автономной* системы ОДУ первого порядка рисуется также поле направлений
- **dfieldplot** – изображение двумерного поля направлений (векторного поля) для системы двух *автономных* ОДУ первого порядка или одного ОДУ первого порядка

График численного решения задачи Коши для системы ОДУ первого порядка с помощью команды odeplot

```
> restart; cond:=x(0)=1,y(0)=2;  
> sys:=diff(x(t),t)=2*y(t)*sin(t)-x(t)-t, diff(y(t),t)=x(t);  
      
$$sys := \frac{d}{dt} x(t) = 2 y(t) \sin(t) - x(t) - t, \frac{d}{dt} y(t) = x(t)$$
  
> F:=dsolve({sys,cond},[x(t),y(t)],numeric):  
> with(plots):  
> p1:=odeplot(F,[t,x(t)],-3..7, color=black, thickness=2, linestyle=3,  
      legend="x(t)");  
> p2:=odeplot(F,[t,y(t)],-3..7,color=green,thickness=2,legend="y(t)");  
> display(p1,p2);
```



Команда DEplot

Двумерные графики решений ОДУ или систем ОДУ

- Команда **DEplot** из пакета **DEtools** строит (двумерные) графики решений ОДУ, а также фазовые портреты и поле направлений. Эта команда аналогична команде **odeplot**, но более функциональна.
- Для дифференциального уравнения порядка выше первого команда **DEplot** рисует только кривые решений ОДУ, а для систем ОДУ первого порядка могут быть нарисованы фазовые портреты с полем направлений.
- В отличие от **odeplot**, команда **DEplot** сама находит численное решение дифференциального уравнения.
- **DEplot(deqns, vars, trange, options)**
- **DEplot(deqns, vars, trange, inits, options)**
- **DEplot(deqns, vars, trange, xrange, yrange, options)**
- **DEplot(deqns, vars, trange, inits, xrange, yrange, options)**
- Основные параметры **DEplot** похожи на параметры **odeplot**:
 - **deqns** – уравнение или список уравнений
 - **vars** – список неизвестных функций;
 - **trange** – диапазон изменения независимой переменной;
 - **inits** – список из списков начальных условий для каждой интегральной кривой;
 - **xrange** и **yrange** – диапазоны изменения первой и второй зависимых переменных (функций); **options** – дополнительные параметры.

Команда DEplot (продолжение)

- Для решения дифференциального уравнения n -го порядка начальные условия можно задавать в компактной форме: $[x0, y0, y'0, y''0, \dots]$, где $x0$ – точка, в которой задаются начальные условия, $y0$ – значение искомой функции в точке $x0$, $y'0, y''0, \dots$ – значения производных: первой, второй и т.д. до $(n-1)$ -го порядка.

Наиболее часто используемые графические параметры команды DEplot

- **linecolor** – цвет линии
- **color** – цвет стрелок
- **arrows=SMALL, MEDIUM, LARGE, LINE** или **NONE** – тип стрелок (значение **NONE** установлено по умолчанию)
- **scene** – определяет, какие зависимости выводить на график: **scene=[x,y]** – двумерный фазовый портрет, **scene=[t,x]** или **scene=[t,x]** – интегральные кривые
- **iterations** – число итераций, необходимое для повышения точности вычислений (по умолчанию это число равно 1)
- **stepsize** – число, равное расстоянию между точками на графике, по умолчанию для интервала $[a,b]$ равно $\text{abs}((b-a))/48$, используется для вывода более гладкой кривой
- **obsrange = true/false** – опция для прерывания вычислений, если график решения выходит за установленный для рисования интервал

График численного решения задачи Коши для системы ОДУ первого порядка с помощью команды DEplot

```
> sys := diff(x(t), t) = 2·y(t)·sin(t) - x(t) - t, diff(y(t), t) = x(t);  
      sys :=  $\frac{d}{dt} x(t) = 2 y(t) \sin(t) - x(t) - t, \frac{d}{dt} y(t) = x(t)$   
=> cond := x(0) = 1, y(0) = 2 :  
=> with(DEtools) :  
=> DEplot([sys], [x(t), y(t)], t = -3 .. 7, x = -10 .. 15, [[cond]], scene = [t, x(t)], linecolor = blue);  
      DEplot([sys], [x(t), y(t)], t = -3 .. 7, x = -20 .. 15, y = -10 .. 20, [[cond]], scene = [t, y(t)],  
      linecolor = green)
```

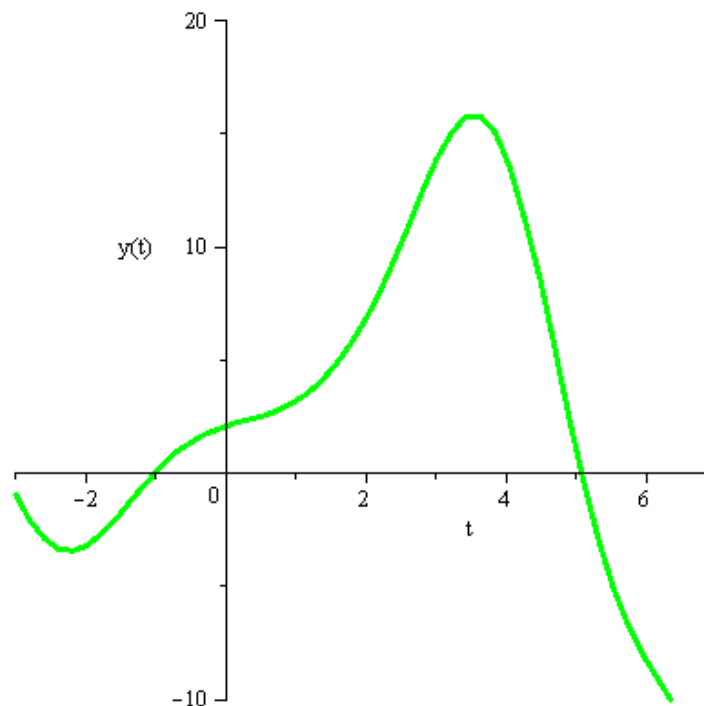
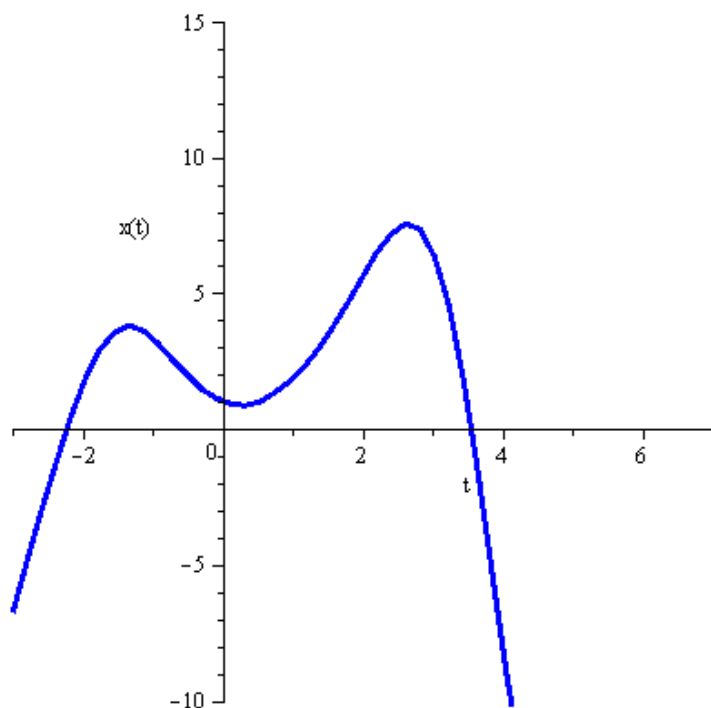


График решения задачи Коши (интегральная кривая) для ОДУ третьего порядка с помощью команды DEplot

```
> restart; with(DEtools):
```

ОДУ 3-го порядка

```
> de3:=diff(y(x),x$3)+x*sqrt(abs(diff(y(x),x)))+x^2*y(x)=0;
```

$$de3 := \frac{d^3}{dx^3} y(x) + x \sqrt{\left| \frac{d}{dx} y(x) \right|} + x^2 y(x) = 0$$

Интегральная кривая для заданных начальных условий

Начальные условия в полной форме записи

```
> DEplot(de3, {y(x)}, x=-4..5, [[y(0)=0,D(y)(0)=1,(D@@2)(y)(0)=1]],  
stepsize=.1, linecolor=black, thickness=2);
```

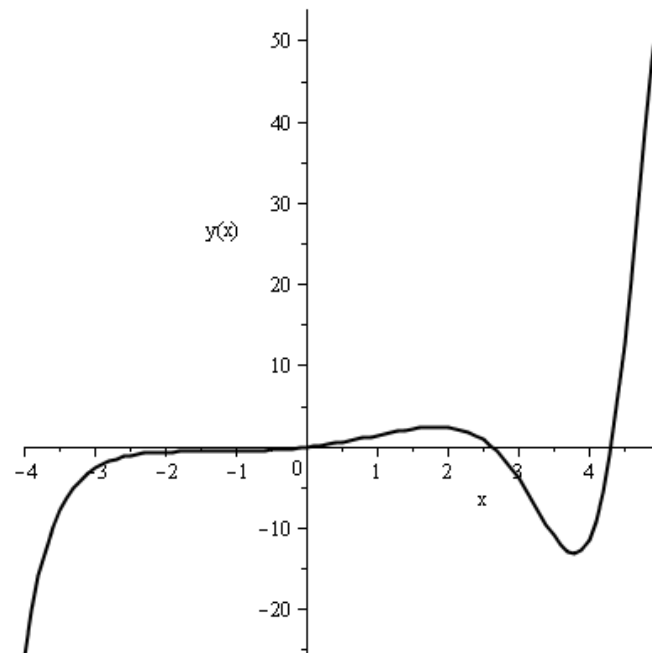


График решения задачи Коши (интегральные кривые и поле направлений) для ОДУ первого порядка с помощью команды DEplot

```
> restart; with(DEtools):
```

ОДУ 1-го порядка

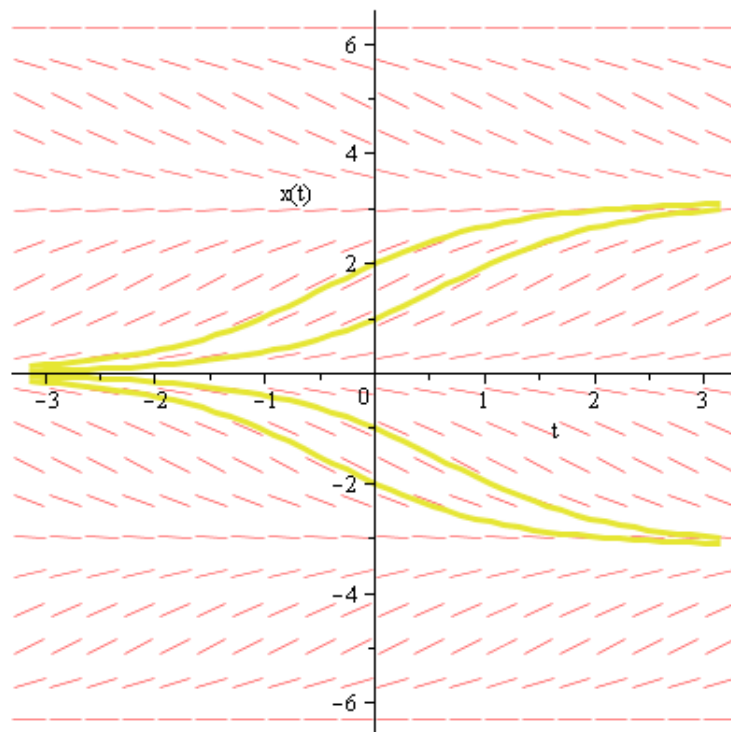
```
> del:=diff(x(t),t)=sin(x(t));
```

$$del := \frac{d}{dt} x(t) = \sin(x(t))$$

Интегральные кривые для нескольких начальных условий в плоскости $(t, x(t))$

Начальные условия в краткой форме записи

```
> DEplot(del, x(t), t=-Pi..Pi, [[0,1],[0,-1],[0,2],[0,-2]], x=-2*Pi..2*Pi,  
arrows=LINE);
```



Команда DEplot3d

Трёхмерные графики решений систем ОДУ

- **DEplot3d(deqns, vars, trange, inits, options)**
- **DEplot3d(deqns, vars, trange, xrange, yrange, inits, options)**

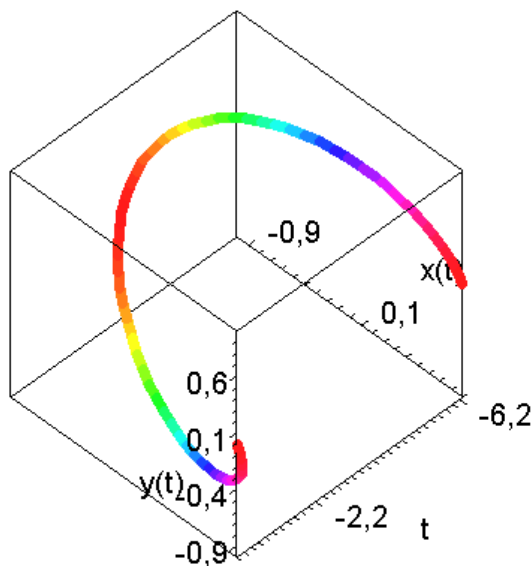
```
> restart:
```

Система двух ОДУ первого порядка

```
> desys:=diff(x(t),t)=-sin(t),diff(y(t),t)=cos(t);
```

$$desys := \frac{d}{dt} x(t) = -\sin(t), \frac{d}{dt} y(t) = \cos(t)$$

```
> with(DEtools):DEplot3d({desys}, {x(t), y(t)}, t = -2*Pi .. 0, [[y(0) = 0, x(0) = 1]],  
    stepsize = .1, linecolor = cos(t));
```



Команда DEplot3d: примеры

```
> restart;
```

Система трех ОДУ первого порядка

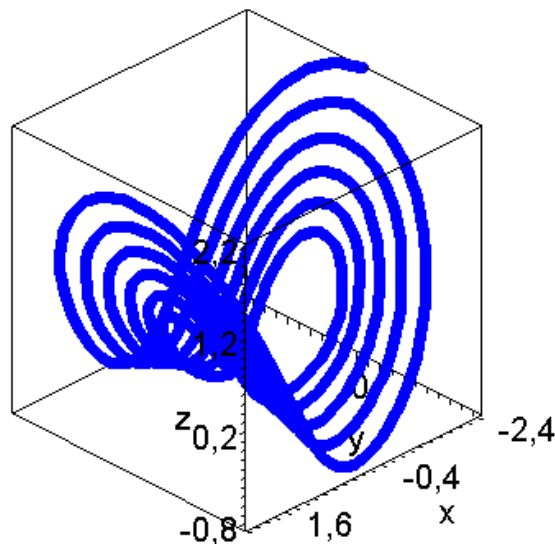
```
> ode3:=diff(x(t),t)=y(t),diff(z(t),t)=-nu*z(t)+x(t)*y(t),  
diff(y(t),t)=-mu*y(t)-(z(t)-1)*x(t)-x(t)^3;  
inits:=[x(0)=1,y(0)=1,z(0)=0];
```

$$\text{ode3} := \frac{d}{dt} x(t) = y(t), \quad \frac{d}{dt} z(t) = -\nu z(t) + x(t) y(t), \quad \frac{d}{dt} y(t) = -\mu y(t) - (z(t) - 1) x(t) - x(t)^3$$

$$\text{inits} := [x(0) = 1, y(0) = 1, z(0) = 0]$$

```
> mu:=-0.1:nu:=0.1: with(DEtools):
```

```
DEplot3d([ode3],[x,y,z],t=0..25,[inits],stepsize=0.05,orientation=[45,60],  
linecolor=blue);
```



Построение фазовых портретов и поля направлений

- Для того чтобы нарисовать весь фазовый портрет (совокупность различных фазовых траекторий), необходимо для каждой фазовой траектории указывать начальные условия.
- Начальные условия можно задавать в компактной форме: $[t_0, x_0, y_0]$, где t_0 – точка, в которой задаются начальные условия, x_0 и y_0 – значения искомых функций в точке t_0 .
- Для *автономной* системы дифференциальных уравнений (т. е. не зависящей явно от t) на фазовом портрете будет построено поле направлений в виде стрелок.

Команды DEplot, phaseportrait и dfieldplot

- С помощью команды **DEplot** можно построить фазовый портрет на плоскости (x, y) для системы двух ОДУ первого порядка $x'(t)=f(x, y, t)$, $y'(t)=g(x, y, t)$, если в параметрах данной команды указать **scene=[x, y]**
- **phaseportrait(deqns, vars, trange, inits, options)** - фазовый портрет для системы двух ОДУ первого порядка или одного ОДУ любого порядка, для автономной системы ОДУ первого порядка рисуется также поле направлений
- **dfieldplot(deqns, vars, trange, xrange, yrange, options)** – поле направлений для системы двух автономных ОДУ первого порядка или одного ОДУ первого порядка. Для команды **dfieldplot** не требуется задавать начальные условия.

Поле направлений нелинейного ОДУ первого порядка с помощью команды DEplot

```
> restart; with(DEtools): nonlin_ode:=diff(y(x),x)=sin(x)/y(x);  
> DEplot(nonlin_ode,y(x),x=-10..10,y=-5..5, stepsize=0.05);
```

- ОДУ первого порядка вида

$$\frac{dy(x)}{dx} = \frac{Q(x, y(x))}{P(x, y(x))} \quad (1)$$

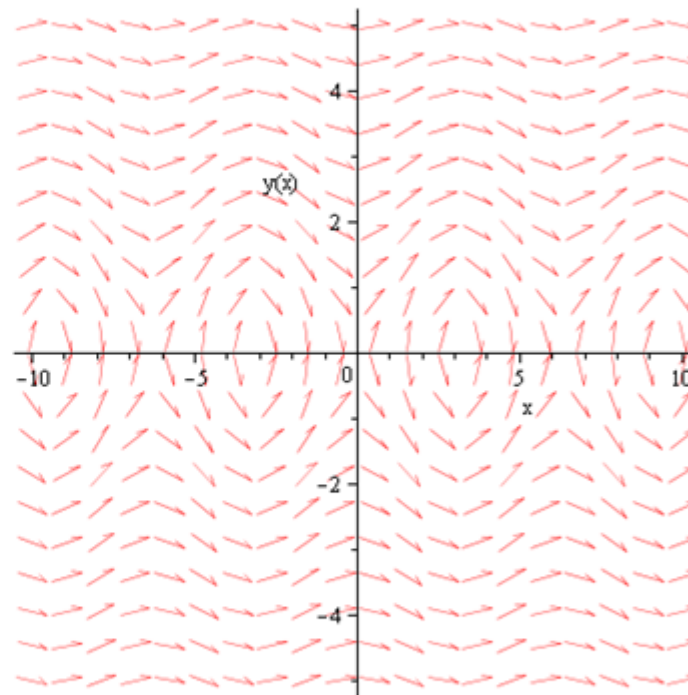
эквивалентно системе двух ОДУ первого порядка вида

$$\begin{cases} \frac{dx(t)}{dt} = P(x(t), y(t)) \\ \frac{dy(t)}{dt} = Q(x(t), y(t)) \end{cases} \quad (2)$$

- Особая точка уравнения (1) или системы (2) – это точка (x_0, y_0) , являющаяся решением системы

$$\begin{cases} P(x, y) = 0, \\ Q(x, y) = 0. \end{cases}$$

$$\text{nonlin_ode} := \frac{d}{dx} y(x) = \frac{\sin(x)}{y(x)}$$



Особые точки

```
> _EnvAllSolutions:=true:solve([sin(x)=0,y=0]);
```

$$\{x = \pi_Z1\sim, y = 0\}$$

Фазовые траектории с полем направлений для нелинейной системы ОДУ первого порядка с заданными начальными условиями (с помощью команды DEplot)

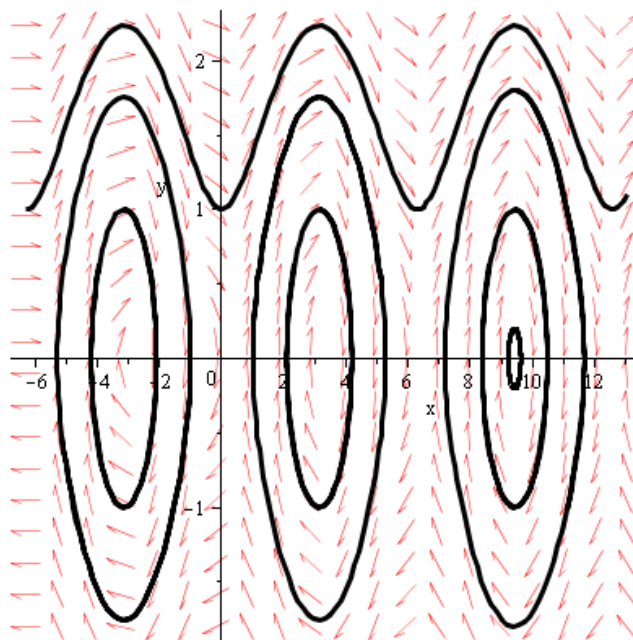
```
> restart; with(DEtools):
```

Эквивалентная система двух ОДУ первого порядка

```
> sys:=diff(x(t),t)=y(t), diff(y(t),t)=sin(x(t));
```

```
> DEplot({sys},[x(t),y(t)], t=0..4*Pi, [[0,1,0], [0,-1,0], [0,Pi,1], [0,-Pi,1],  
[0,3*Pi,0.2], [0,3*Pi,1], [0,3*Pi,1.8], [0,-2*Pi,1]],stepsize=0.1, linecolor=  
black);
```

$$\text{sys} := \frac{d}{dt} x(t) = y(t), \frac{d}{dt} y(t) = \sin(x(t))$$



Фазовые траектории без поля направлений для нелинейной системы ОДУ первого порядка с заданными начальными условиями (с помощью команды DEplot)

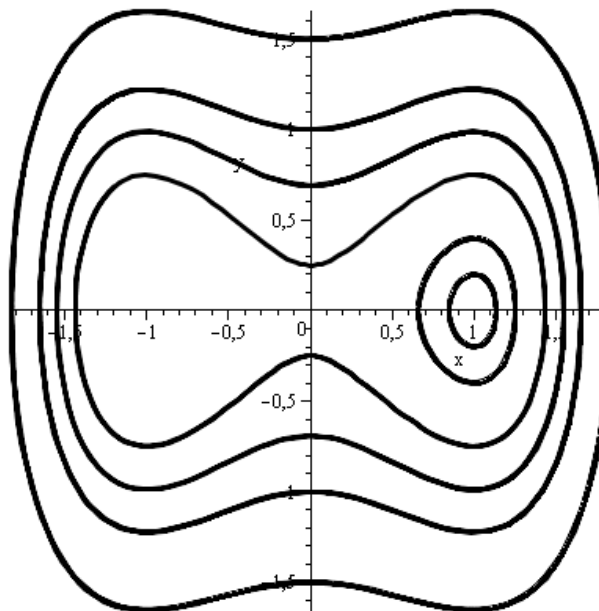
```
> restart; with(DEtools):
```

Нелинейная автономная система двух ОДУ первого порядка

```
> sys:=diff(x(t),t)=y(t), diff(y(t),t)=x(t)-x(t)^3;
```

$$\text{sys} := \frac{d}{dt} x(t) = y(t), \quad \frac{d}{dt} y(t) = x(t) - x(t)^3$$

```
> DEplot({sys}, [x(t),y(t)], t=0..20, [[0,1,0.2], [0,0,1], [0,1,0.4], [0,1,0.75],  
[0,0,1.5], [0,-0.1,0.7]], stepsize=0.1, arrows=none, linecolor=black);
```

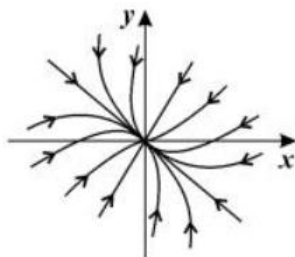


Фазовые портреты линейных автономных систем дифференциальных уравнений первого порядка

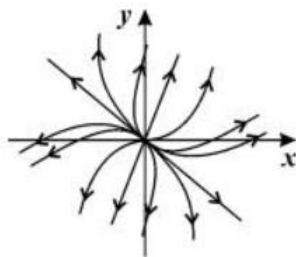
Дана линейная динамическая система $\begin{cases} \frac{dx}{dt} = ax + by, \\ \frac{dy}{dt} = cx + dy \end{cases}$ или уравнение $\frac{dy}{dx} = \frac{cx + dy}{ax + by}$

Особая точка: $\{x=0, y=0\}$, если $\det(A) \neq 0$ или все точки прямой $ax+by=0$, если $\det(A)=0$ (т.е. $a/b=c/d$), где A – матрица системы.

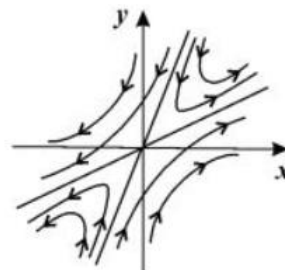
Характеристическое уравнение: $\begin{vmatrix} a-\lambda & b \\ c & d-\lambda \end{vmatrix} = 0$ Матрица системы: $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$



Устойчивый узел.
(λ_1, λ_2 действительны и отрицательны)

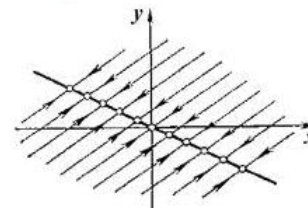


Неустойчивый узел.
(λ_1, λ_2 действительны и положительные)

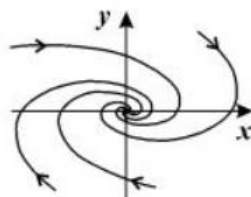


Седло.
(λ_1, λ_2 - действительны и разных знаков)

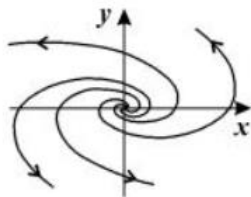
Непростая особая точка



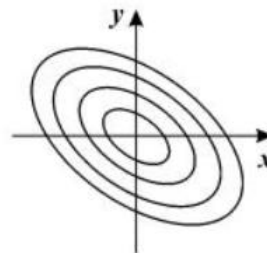
$\lambda_1=0, \lambda_2<0$



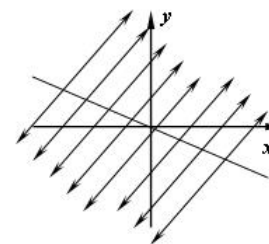
Устойчивый фокус
(λ_1, λ_2 - комплексны,
 $\text{Re } \lambda_{1,2} < 0$)



Неустойчивый фокус
(λ_1, λ_2 - комплексны,
 $\text{Re } \lambda_{1,2} > 0$)



Центр.
(λ_1, λ_2 - чисто мнимые)

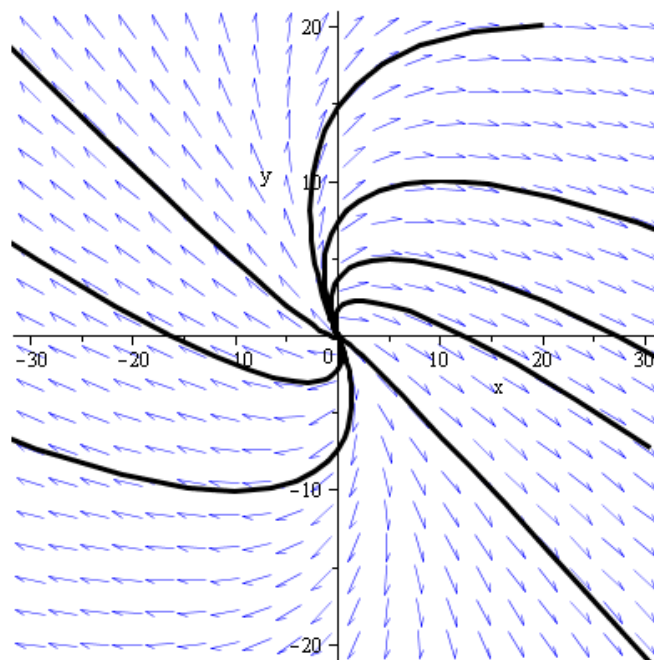


$\lambda_1=0, \lambda_2>0$

Фазовые траектории и поле направлений для линейной автономной системы ОДУ первого порядка с заданными начальными условиями (с помощью команды phaseportrait)

```
> restart; with(DEtools):  
> sys:=diff(x(t),t)=3*x(t)+y(t), diff(y(t),t)=-x(t)+y(t);  
> phaseportrait([sys],[x(t),y(t)],t=-10..10, [[0,1,-2],[0,-3,-3],[0,-2,4],  
[0,5,5],[0,5,-3],[0,-5,2],[0,5,2],[0,-1,2]], x=-30..30,y=-20..20,  
stepsize=.1, colour=blue,linecolor=black);
```

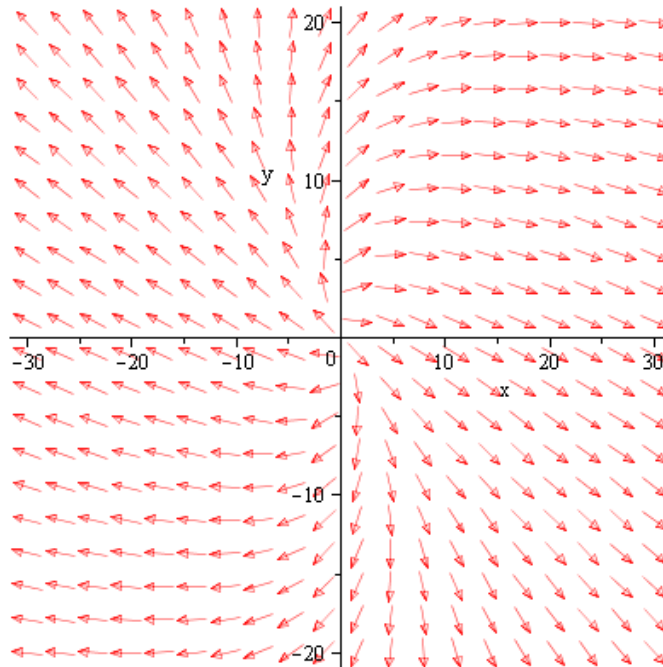
$$sys := \frac{d}{dt} x(t) = 3x(t) + y(t), \frac{d}{dt} y(t) = -x(t) + y(t)$$



Поле направлений для линейной системы ОДУ первого порядка (с помощью команды dfieldplot)

```
> restart; with(DEtools):  
> sys:=diff(x(t),t)=3*x(t)+y(t), diff(y(t),t)=-x(t)+y(t);  
> dfieldplot([sys],[x(t),y(t)],t=-10..10, x=-30..30,y=-20..20, stepsize=.1,  
arrows = SLIM);
```

$$\text{sys} := \frac{d}{dt} x(t) = 3x(t) + y(t), \frac{d}{dt} y(t) = -x(t) + y(t)$$



Решение уравнений в частных производных

- Команда `pdsolve`
- Пакет `PDEtools`: обзор команд
- Команда `PDEplot`

Команда pdsolve

Решение уравнения или системы ДУЧП (дифференциальных уравнений в частных производных)

- **pdsolve(PDE, f, other_options)**
- **pdsolve(PDE_system, funcs, other_options)**
- **pdsolve(PDE_or_PDE_system, conds, type=numeric, other_options)**

Проверка найденного решения

- **pdetest(sol,pde)**

```
> PDE := x*diff(f(x,y),y)-y*diff(f(x,y),x) = 0;
```

$$PDE := x \left(\frac{\partial}{\partial y} f(x, y) \right) - y \left(\frac{\partial}{\partial x} f(x, y) \right) = 0$$

```
> sol:=pdsolve(PDE);
```

$$sol := f(x, y) = _F1(x^2 + y^2)$$

_F1 - произвольная функция

Проверка решения

```
> pdetest(sol,PDE);
```

Команда pdsolve: использование опций

Решение задачи теплопроводности

```
> heat:=diff(u(x,t),t)=diff(u(x,t),x,x);
```

$$heat := \frac{\partial}{\partial t} u(x, t) = \frac{\partial^2}{\partial x^2} u(x, t)$$

Опция поиска решения в виде произведения

```
> pdsolve(heat,HINT=`*`);
```

$$(u(x, t) = _F1(x) _F2(t)) \&where \left[\left\{ \frac{d^2}{dx^2} _F1(x) = _c1 _F1(x), \frac{d}{dt} _F2(t) = _c1 _F2(t) \right\} \right]$$

Зададим явное представление для решения

```
> sol:=pdsolve(heat,HINT=X(x)*T(t),build);
```

$$sol := u(x, t) = e^{\sqrt{-c_1} x} _C1 e^{-c_1 t} _C2 + \frac{_C1 e^{-c_1 t} _C3}{e^{\sqrt{-c_1} x}}$$

Найдем частное решение для заданных параметров

```
> const:={_c[1]=-4, _C1=I/2, _C2=-1, _C3=1};
```

```
> simplify(eval(rhs(sol),const));
```

$$e^{-4t} \sin(2x)$$

Пакет PDEtools

- **PDEtools** – пакет для работы с дифференциальными уравнениями в частных производных (ДУЧП), содержит команды для исследования и нахождения аналитических решений ДУЧП, визуализации решений ДУЧП и т. д.

> *with(PDEtools)*

*[CanonicalCoordinates, ChangeSymmetry, CharacteristicQ, D_Dx,
DeterminingPDE, Hk, FromJet, InfinitesimalGenerator, Infinitesimals,
InvariantSolutions, InvariantTransformation, Invariants, PDEplot,
ReducedForm, SimilaritySolutions, SimilarityTransformation, SymmetryTest,
SymmetryTransformation, TWSolutions, ToJet, build, casesplit, charstrip,
dchange, dcoeffs, declare, diff_table, difforder, dpolyform, dsubs, mapde,
separability, splitstrip, splitsys, undeclare]*

Пакет PDEtools: команда PDEplot

- **PDEplot(PDE, inits, srange, options)** – решение и построение графика ДУЧП первого порядка (линейного или нелинейного)

Решение нелинейного волнового уравнения

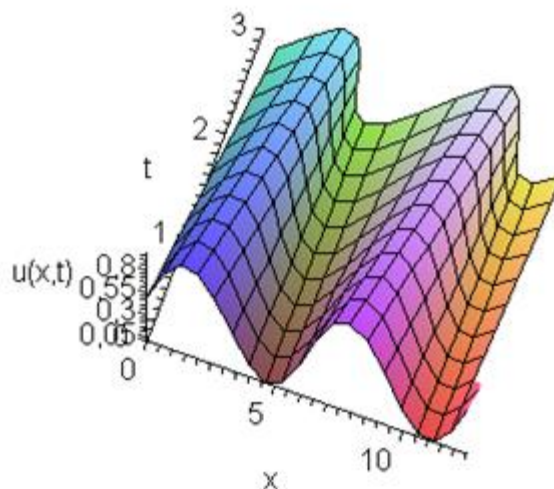
```
> pde :=  $\frac{\partial}{\partial t} u(x, t) + u(x, t) \cdot \frac{\partial}{\partial x} u(x, t);$ 
```

$$pde := \frac{\partial}{\partial t} u(x, t) + u(x, t) \left(\frac{\partial}{\partial x} u(x, t) \right)$$

```
> ini := [s, 0, 0.5 + 0.5 · sin(s) ] :
```

```
> with(PDEtools) :
```

```
> PDEplot(pde, x = 0 .. 4 · π, t = 0 .. 3, ini, s = 0 .. 4 · π, style = patch, orientation  
= [ -70, 15 ], axes = normal )
```



Пакет PDEtools: некоторые другие команды

- **dchange(tr, expr)** – замена переменных, где **tr** – список уравнений вида **old=new**
- **dsubs(deriv=a, expr)** – подстановка выражений для производных

```
> with(PDEtools):
```

```
> PDE :=  $\frac{\partial}{\partial x} f(x, y) + g(x, y) + \frac{\partial}{\partial y} f(x, y) = 0$ 
```

$$PDE := \frac{\partial}{\partial x} f(x, y) + g(x, y) + \frac{\partial}{\partial y} f(x, y) = 0$$

```
> tr := {x = r + s, y = r - s}; dchange(tr, PDE)
```

$$tr := \{x = r + s, y = r - s\}$$

$$\frac{\partial}{\partial r} f(r, s) + g(r, s) = 0$$

```
> eq1 :=  $\frac{d}{dx} f(x) = f(x)$ 
```

$$eq1 := \frac{d}{dx} f(x) = f(x)$$

```
> expr :=  $\frac{d^2}{dx^2} f(x) - f(x)$ 
```

$$expr := \frac{d^2}{dx^2} f(x) - f(x)$$

Обзор пакетов Maple

- Упомянувшиеся в данном курсе
- Некоторые математические пакеты по темам
- Другие пакеты

Рассмотренные пакеты по темам

Тема	Названия пакетов	Устаревшая версия
Двумерная и трехмерная графика, графические объекты	plots, plottools	
Линейная алгебра	LinearAlgebra	linalg
Векторный анализ	VectorCalculus	linalg
Исследование ОДУ	DEtools	
Исследование ДУ в частных производных	PDEtools	
Интегральные преобразования	inttrans	
Создание maplets	Maplets	
Учебные вычисления	Student и его подпакеты Student[Calculus1] Student[LinearAlgebra] Student[MultivariateCalculus] Student[Precalculus] Student[VectorCalculus]	student

Некоторые математические пакеты по темам

Тема	Названия пакетов	Устаревшая версия
Статистика	Statistics	stats
Операции с полиномами	PolynomialTools	polytools
Теория графов	GraphTheory	networks
Степенные разложения	powseries	
Задачи оптимизации	Optimization (числ. решение), simplex (лин. оптимизация)	
Теория чисел	numtheory	
Математическая логика	Logic (logic)	
Аппроксимация функций	numapprox	
Комбинаторика	combinat, combstruct	
Финансовая математика	finance	
Евклидова геометрия	geometry, geom3d	
Тензорное исчисление	tensor	
Вариационное исчисление	VariationalCalculus	

Некоторые другие пакеты

Тема	Названия пакетов
Работа с аудиофайлами	AudioTools
Битовые операции	Bits
Перевод процедур и кода Maple на другие языки программирования	codegen, CodeGeneration
Создание контекстных меню	ContextMenu
Использование баз данных	Database
Работа с документами Maple	DocumentTools
Работа с данными в формате Excel	ExcelTools
Работа с файлами	FileTools
Обработка изображений	ImageTools
Создание библиотек и работа с ними	LibraryTools
Работа с электронными таблицами	Spread
Использование XML-документов	XMLTools
Работа со строками	StringTools
Сетевой обмен данными	Sockets
Взаимодействие с Matlab	Matlab