Доля П.Г.
Харьковский Национальный Университет
механико – математический факультет
2015 г.

Mathematica для математиков.
Часть 3. Реализация основных понятий математического анализа.

В системе *Mathematica* реализованы все основные понятия, используемые в математическом анализе: пределы, производные, интегралы, ряды. Предполагается, что читатель знаком с этими понятиями. В данной главе представлен обзор основных функций системы, которые выполняют операции математического анализа символьно и численно.

Примеры, приводимые в пособии, проверялись в версии пакета *Mathematica* 9.0. В предыдущих версиях многие примеры работать не будут. Особенности некоторых функций старых версий мы иногда описываем отдельно.

Оглавление

Глава 3. Реализация основных понятий математического анализа в системе Mathematica.

3.1 Вычисление пределов

Для вычисления предела используется команда `Limit`

$$\mathbf{Limit}\left[\frac{\mathbf{Sin}[x]}{x}, x \to 0\right]$$

1

Первый аргумент - функция, для которой находится предел. Второй аргумент определяет переменную и значение, к которому она приближается. Запись `x→0` читается, как "x стремится к нулю". Вычисление предела можно записать в традиционном виде.

$$\mathbf{Limit}[f[x], x \to x_0]//\mathbf{TraditionalForm}$$

$$\lim_{x \to x_0} f(x)$$

Выполним несколько примеров.

$$\mathbf{Limit}[\frac{\sqrt{1+x}-\sqrt{1-x}}{x}, x \to 0]$$

1

$$\mathbf{Limit}[\frac{(1+x)^2-1}{x}, x \to 0]$$

2

$$\mathbf{Limit}[\frac{10^x-1}{x}, x \to 0]$$

Log[10]

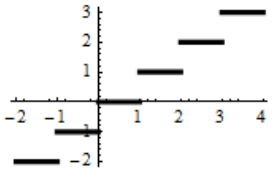
$$\mathbf{Limit}[x\mathbf{Log}[x], x \to 0]$$

0

Можно вычислять пределы, когда переменная стремится к бесконечности

$$\mathbf{Limit}\left[\frac{(x+1)^2}{x^2}, x \to \infty\right]$$
1
$$\mathbf{Limit}[\frac{3^x - 3^{-x}}{3^x + 3^{-x}}, x \to -\infty]$$
-1
$$\mathbf{Limit}[\left(1 + \frac{z}{x}\right)^x, x \to \mathbf{Infinity}]$$
e^z
$$\mathbf{Limit}[\left(\frac{2n+z}{2n-z}\right)^n, n \to \infty]$$
e^z
$$\mathbf{Limit}[e^{-e^{-e^x}+\frac{1}{x}+x}(-1 + e^{e^{-e^x}+e^{-x}+e^{-x^2}}), x \to \infty]$$
1
Можно вычислять односторонние пределы. Для этого нужно указать направление, используя опцию `Direction`. Следующая команда вычисляет левый предел (т.е. двигаясь из нуля направо или в положительном направлении)
$$\mathbf{Limit}\left[\mathbf{Tan}[x], x \to \frac{\pi}{2}, \mathbf{Direction} \to 1\right]$$
∞
В следующем примере происходит приближение справа.
$$\mathbf{Limit}\left[\mathbf{Tan}[x], x \to \frac{\pi}{2}, \mathbf{Direction} \to -1\right]$$
$-\infty$
Заметим, что значение опции `Direction` определяет, куда вы хотите передвигаться, а не откуда вы идете. Таким образом, отрицательное значение указывает на движение в отрицательном направлении, т.е. приближении справа. Важен только знак, не величина (вместо единицы можно написать любое число).

Направление приближения существенно при вычислении пределов разрывных функций. Например рассмотрим функцию вычисления целой части числа (наибольшего целого, не превосходящего данное)
$$\mathbf{Plot}[\mathbf{Floor}[x], \{x, -2, 4\}]$$



Вычислим ее пределы в точке 2.
$$\mathbf{Limit}[\mathbf{Floor}[x], x \to 2]$$
2
$$\mathbf{Limit}[\mathbf{Floor}[x], x \to 2, \mathbf{Direction} \to 1]$$
1

$$\mathbf{Limit}[\mathbf{Floor}[x], x \to 2, \mathbf{Direction} \to -1]$$

2

При стремлении переменной к конечному пределу по умолчанию вычисляется предел справа (Direction $\to -1$).

Предел полезен для проверки вычисления значения функции в окрестности точек разрыва. Например вычислим значение выражения $\frac{1-\cos x^2}{x^4}$ вблизи нуля.

$$\frac{\mathbf{1 - Cos}[x^2]}{x^4} /. \{\{x \to \mathbf{0.01}\}, \{x \to \mathbf{0.00001}\}, \{x \to \mathbf{0.0000100000000000000000}\}\}$$

{0.5, 0., 0.50000000000000000}

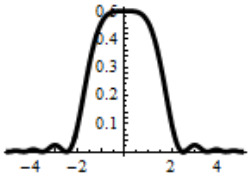
В точке x=0.00001 значение функции вычислено неверно. Это произошло из-за потери точности. В той же точке, но при более высокой точности задания переменной x (третье значение в списке) происходит вычисления значения функции с более высокой точностью. Проверка значения предлом дает 0.5

$$\mathbf{Limit}\left[\frac{\mathbf{1 - Cos}[x^2]}{x^4}, x \to 0\right]$$

$\frac{1}{2}$

График выражения также проясняет ситуацию.

$$\mathbf{Plot}[\frac{\mathbf{1 - Cos}[x^2]}{x^4}, \{x, -5, 5\}]$$



Можно вычислять повторные пределы

$$\mathbf{Limit}[\mathbf{Limit}[x^2y^2 - 2xy^5 + 3y, x \to 2], y \to 3]$$

-927

В некоторых случаях важен порядок вычисления пределов.

$$\mathbf{Limit}[\mathbf{Limit}\left[\frac{x^2y}{x^2 + y^2}, x \to \infty\right], y \to \infty]$$

∞

$$\mathbf{Limit}[\mathbf{Limit}\left[\frac{x^2y}{x^2 + y^2}, y \to \infty\right], x \to \infty]$$

0

Для функций нескольких переменных можно вычислять пределы по множеству

$$\mathbf{Limit}[\frac{xy}{x^2 + y^2} /. x \to ty, y \to 0]$$

$\frac{t}{1+t^2}$

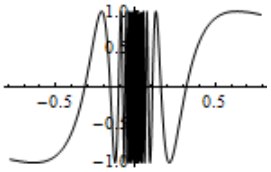
В тех случаях, когда предел не может быть вычислен, функция `Limit` возвращает интервал изменения значения функции в котором значение функции находится.

$$\mathbf{Limit}[\mathbf{Sin}[\mathbf{1}/\boldsymbol{x}], \boldsymbol{x} \to \mathbf{0}]$$

Interval[{−1,1}]

Чтобы понять, что это значит, построим график функции

$$\mathbf{Plot}[\mathbf{Sin}\left[\frac{\mathbf{1}}{\boldsymbol{x}}\right], \{\boldsymbol{x}, -\boldsymbol{\pi}/\mathbf{4}, \boldsymbol{\pi}/\mathbf{4}\}]$$



Если вычисляется предел функции, о которой нет определенной информации, то возвращается необработанный ввод.

$$\mathbf{Limit}[\boldsymbol{x}\boldsymbol{f}[\boldsymbol{x}], \boldsymbol{x} \to \mathbf{0}]$$

$\mathrm{Limit}[xf[x], x \to 0]$

Однако имеется опция `Analytic ->True`, которая гооворит функции `Limit` делать предположения о том, что функции общего вида, о которых нет явной информации, следует считать аналитическими (в частности считать их непрерывными с непрерывными производными любого порядка). Тогда

$$\mathbf{Limit}[\boldsymbol{x}\,\boldsymbol{f}[\boldsymbol{x}], \boldsymbol{x} \to \mathbf{0}, \mathbf{Analytic} \to \mathbf{True}]$$

0

При вычислении пределов в функции `Limit` с помощью опции `Assumptions` можно указывать предположения относительно символьных переменных.

$$\mathbf{Limit}[\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x} \to \infty, \mathbf{Assumptions} \to \boldsymbol{a} < 0]$$

0

$$\mathbf{Limit}[\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x} \to \infty, \mathbf{Assumptions} \to \boldsymbol{a} > 0]$$

∞

$$\mathbf{Limit}[\boldsymbol{a}^{\boldsymbol{x}}/\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x} \to \infty, \mathbf{Assumptions} \to \mathbf{0} < \boldsymbol{a} < 1]$$

0

Вы можете использовать `Limit` для вычисления производных «реальных» функций

$$\boldsymbol{f}[\mathrm{x\_}] = \boldsymbol{x}^{\mathbf{4}};$$

$$\mathbf{Limit}[\frac{\boldsymbol{f}[\boldsymbol{x} + \mathbf{dx}] - \boldsymbol{f}[\boldsymbol{x}]}{\mathbf{dx}}, \mathbf{dx} \to \mathbf{0}]$$

$4x^3$

$$\mathbf{Limit}[\frac{\boldsymbol{f}[\boldsymbol{x} + \mathbf{2}\boldsymbol{h}] - \mathbf{2}\boldsymbol{f}[\boldsymbol{x} + \boldsymbol{h}] + \boldsymbol{f}[\boldsymbol{x}]}{\boldsymbol{h}^{\mathbf{2}}}, \boldsymbol{h} \to \mathbf{0}]$$

$12x^2$

или аналитических функций общего вида

$$\mathbf{Clear}[\boldsymbol{f}];$$

$$\mathbf{Limit}[\frac{\boldsymbol{f}[\boldsymbol{x} + \boldsymbol{h}] - \boldsymbol{f}[\boldsymbol{x}]}{\boldsymbol{h}}, \boldsymbol{h} \to \mathbf{0}, \mathbf{Analytic} \to \mathbf{True}]$$

$f'[x]$

$$\mathbf{Limit}[\frac{\boldsymbol{f}[\boldsymbol{x}+\mathbf{2}\boldsymbol{h}]-\mathbf{2}\boldsymbol{f}[\boldsymbol{x}+\boldsymbol{h}]+\boldsymbol{f}[\boldsymbol{x}]}{\boldsymbol{h}^{\mathbf{2}}},\boldsymbol{h}\to\mathbf{0},\mathbf{Analytic}\to\mathbf{True}]$$

$f''[x]$

Сумма бесконечного числового ряда определяется как предел частичных сумм, например, $\sum_{k=1}^{\infty}\frac{1}{k(k+1)}=\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k(k+1)}$. Тогда

$$\mathbf{sm}=\mathbf{Sum}[\frac{\mathbf{1}}{\boldsymbol{k}(\boldsymbol{k}+\mathbf{1})},\{\boldsymbol{k},\mathbf{1},\boldsymbol{n}\}]$$

$$\mathbf{Limit}[\mathbf{sm},\boldsymbol{n}\to\infty]$$

$$1-\frac{1}{1+n}$$

1

Определенный интеграл Римана является пределом интегральных сумм. Пусть это будет

$$\int_a^b f(x)\,dx=\frac{b-a}{n}\lim_{n\to\infty}\sum_{k=0}^{n-1}f\left(a+k\frac{b-a}{n}\right)$$

Выражение, стоящее в правой части, можно записать в виде функции

$$\mathbf{rs}:=\mathbf{Function}[\{\boldsymbol{f},\boldsymbol{x},\boldsymbol{a},\boldsymbol{b},\boldsymbol{n}\},\frac{\boldsymbol{b}-\boldsymbol{a}}{\boldsymbol{n}}\mathbf{Sum}[\boldsymbol{f}[\boldsymbol{a}+\boldsymbol{k}\frac{(\boldsymbol{b}-\boldsymbol{a})}{\boldsymbol{n}}],\{\boldsymbol{k},\mathbf{0},\boldsymbol{n}-\mathbf{1}\}]];$$

Используя созданную функцию, можно вычислять определенный интеграл. Например, вычислим следующий интеграл $\int_0^1 x^3\,dx$.

$$\mathbf{rl}=\mathbf{rs}[\#^{\mathbf{3}}\&,\boldsymbol{x},\mathbf{0},\mathbf{1},\boldsymbol{n}]$$

$$\mathbf{Limit}[\mathbf{rl},\boldsymbol{n}\to\infty]$$

$$\frac{(-1+n)^2}{4n^2}$$

$$\frac{1}{4}$$

Или вычислим интеграл $\int_0^1 e^{ax}\,dx$.

$$\mathbf{rl}=\mathbf{rs}[\mathbf{Exp}[\boldsymbol{a}\#]\&,\boldsymbol{x},\mathbf{0},\mathbf{1},\boldsymbol{n}];$$

$$\mathbf{Limit}[\mathbf{rl},\boldsymbol{n}\to\infty]$$

$$\frac{-1+e^{a}}{a}$$

Неопределенный интеграл можно вычислять как определенный с переменным верхним пределом. Например вычислим $\int_0^x t^3\,dt$

$$\mathbf{rl}=\mathbf{rs}[\#^{\mathbf{3}}\&,\boldsymbol{t},\mathbf{0},\boldsymbol{x},\boldsymbol{n}];$$

$$\mathbf{Limit}[\mathbf{rl},\boldsymbol{n}\to\infty]$$

$$\frac{x^4}{4}$$

Несобственный интеграл с бесконечными пределами является пределом определенного интеграла с конечными пределами, когда один (или оба) из пределов стремится к бесконечности. Например $\int_0^{\infty} e^{-3x}\,dx = \lim_{a\to\infty}\int_0^{a} e^{-3x}\,dx$. Тогда

```
sum0a = rs[Exp[-3#]&, x, 0, a, n];
int0a = Limit[sum0a, n → ∞];
Limit[imt0a, a → ∞]
```

$$\frac{1}{3}$$

Интеграл от неограниченных функций тоже вычисляется с использованием предельного перехода

```
Assuming[0 < ep < 1, intgr = ∫_ep^1 1/√x dx]
```

$$2 - 2\sqrt{\text{ep}}$$

Здесь функция `Assuming` накладывает ограничение на переменную `ep`, используемую в интеграле. В данном случае выражение (интеграл), стоящее во втором аргументе, вычисляется в предположении, что $0 < ep < 1$. После этого вычисляем предел

```
Limit[intgr, ep → 0]
```

2

Или

```
Assuming[0 < one < 1, intgr = ∫_0^one x/√(1 - x^2) dx];
Limit[intgr, one → 1]
```

1

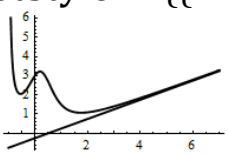
Конечно, функция `Limit` не предназначена для вычисления производных и интегралов. Для этого в системе есть более «умные» функции. С ними вы познакомитесь немного позже. Функция `Limit` нужна, например, для определения асимптот функций. Вот как это можно сделать.

```
f[x_] = (x^4 + 2x + 3)/(2x^3 + x^2 + 1);
a = Limit[f[x]/x, x → ∞]
b = Limit[f[x] - ax, x → ∞]
```

$$\frac{1}{2}$$

$$-\frac{1}{4}$$

```
Plot[{f[x], ax + b}, {x, -1, 7}, PlotStyle → {{Black, Thickness[0.01]}}]
```



Встроенная функция `Limit` вычисляет пределы, используя аналитические методы. Кроме нее имеется внешняя функция `NLimit`, выполняющая поиск предела численно с помощью последовательности значений, приближающихся к указаному предельному значению. Результат последовательного вычисления передается в процедуру, которая аппроксимирует полученные значения и возвращает предел.

Needs["**NumericalCalculus`**"]

(В Mathematica 5 нужно загружать пакет `<<NumericalMath`NLimit`)`

$$\mathbf{NLimit}\left[\frac{\mathbf{Sin}[x]}{x}, x \to 0\right]$$

1

$$\mathbf{NLimit}\left[\frac{2x^3 + \mathbf{Sin}[x]}{5x^3 + \mathbf{Log}[x]}, x \to \infty\right]$$

0.4

Заметим, что функция `Limit` может вернуть неправильный результат, если предельное значение указано приближенно, а не точно (числа с десятичной точкой рассматриваются как приближенные).

$$s = \mathbf{Limit}[\mathbf{Log}[1 - (\mathbf{Log}[\mathbf{Exp}[z]/z - 1] + \mathbf{Log}[z])/z]/z, z \to 100.]$$

$-\infty$

Но ответ получается правильный, если входные значения заданы точно.

$$s = \mathbf{Limit}[\mathbf{Log}[1 - (\mathbf{Log}[\mathbf{Exp}[z]/z - 1] + \mathbf{Log}[z])/z]/z, z \to 100]$$

$$\frac{1}{100}\text{Log}[1 - \frac{1}{100}\text{Log}[-100 + e^{100}]]$$

Тем не менее вычисления полученного результата с недостаточной точностью тоже неверны

$N[s]$

$N[s, 20]$

Indeterminate

-1.0000000000000000000

3.2 Дифференцирование

3.2.1 Создание и использование функций

Операция дифференцирования применяется к функциям. Поэтому здесь будет уместно напомнить некоторые основные способы создания функций.

Простейший способ состоит в выполнении команды

$y[\mathbf{x_}] := $ *выражение*

или

$y[\mathbf{x_}] = $ *выражение* (без двоеточия)

Здесь *y* – имя функции (такое как Sin для функции Sin[x]). Символ подчеркивания в левой части говорит о том, что переменная *x* в правой части выражения является локальной. Знак **:=** представляет отложенное присваивание, которое выполняется в тот момент, когда происходит

конкретное вычисление значения функции. Если используется знак = , то выполняется создание функции немедленно.

Общий способ создания функции пользователя состоит в использовании ключевого слова `Function`.

```
funcname = Function[x, body]
```

Эта форма предназначена для определения функции с одним формальным параметром (переменной) *x*. Здесь `funcname` имя функции, такое как Sin, Log и т.д., *body* – выражение, определяющее правило вычисления. Например

$$\boldsymbol{z = \text{Function}[x, x^2]};$$

$$\boldsymbol{z[2]}$$

4

Идентификатор *z*, стоящий слева, является именем функции, первое вхождение *x* в правой части обозначает имя аргумента, затем следует выражение, описывающее правило вычисления значения функции. Выражение может состоять из нескольких выражений, разделенных точкой с запятой. Результат вычисления последнего выражения в такой последовательности будет результатом вычисления функции.

$$\boldsymbol{\text{f1} = \text{Function}[x, g = x^2; g * \text{Sin}[x]]};$$

Можно не определять имя функции при ее создании

$$\boldsymbol{\text{Function}[x, x^2]};$$

Тогда можно использовать знак % вместо имени функции (% представляет результат выполнения последней операции в системе). Например,

$$\%[\boldsymbol{n}]$$

$$n^2$$

Функцию можно определить без указания имен ее аргументов. В таком случае для обращения к ним следует использовать значки #1, #2 и т.д.

```
funcname=Function[body]
```

где `funcname` имя функции, *body* – выражение, определяющее правило вычисления. Например,

$$\boldsymbol{\text{f2} = \text{Function}[\#1^2]}$$

$$\#1^2\&$$

Идентификатор #1 заменяет имя первого (и единственного) аргумента. Тогда в выражениях можно использовать `f2[x]` или вместо *x* подставлять конкретные значения `f2[2]`.

Синонимом основного определения функции является ее постфиксное определение в виде

$$\boldsymbol{\text{funcname} = (\#1^2)}\&$$

$$\#1^2\&$$

Аргумент при определении функции обозначается # (или #1, #2 и т.д.). Само имя функции обозначается #0, ## используется для обозначения всего списка аргументов, ##n обозначает списка аргументов, начиная с n-того. Значок **&** используется вместо ключевого слова `Function` после указания тела функции. Таким образом, последнее определение эквивалентно постфиксному использованию слова `Function`.

$$\mathbf{funcname} = \overbrace{\left(\#1^2\right)}^{\&}\text{//Function}$$

Для определения функции с несколькими аргументами можно использовать форму `Function[{x1, x2,…},body]` с использованием списка формальных параметров `{x1,x2,…}`. Можно использовать и простую форму, например

$\boldsymbol{z[x\_, y\_] = x^2 + y^2;}$

$\boldsymbol{z[2, 3]}$

13

Есть еще способ создания функции с использованием символа отображения (стрелки)

$\boldsymbol{z = x \to x^3}$

$\text{Function}[x, x^3]$

Здесь стрелка особая! Чтобы правильно ее набрать надо последовательно нажать комбинацию клавиш $Esc - fn - Esc$.

Для создания функций без побочных эффектов удобно использовать ключевое слово `Module`. Имена аргументов функции указываются с символом подчеркивания в левой части операции присваивания, а в правой используется список локальных имен модуля, за которым следуют выражения

```
Module[{имя1,имя2,...},выражение1;выражение2;...]
```

Например

$\boldsymbol{f[\text{x\_}] := \text{Module}[\{t\}, t = (1 + x)\text{\^{}}3; t = \text{Expand}[t]]}$

$\boldsymbol{f[s]}$

$1 + 3s + 3s^2 + s^3$

Имена переменных, использованных в первом списке справа, будут локальными переменными модуля и не будут конфликтовать с такими же именами в глобальной области рабочего пространства системы. При этом они могут рассматриваться не только как имена переменных, но и как имена локальных функций. С использование функции `Module` могут конструироваться функции многих переменных

В *Mathematica* есть несколько способов применить функцию к выражению. Это обычные квадратные скобки f[x], префикс f@x и постфикс x // f. Например

$\boldsymbol{\{\text{Sin}\left[\frac{\pi}{6}\right],\ \text{Sin@}\frac{\pi}{4},\ \frac{\pi}{3}\text{//Sin}\}}$

$\left\{\frac{1}{2}, \frac{1}{\sqrt{2}}, \frac{\sqrt{3}}{2}\right\}$

Префиксное и постфиксное использование имени функции используется для функций одной переменной. Для функции двух переменных возможно инфиксное (между аргументами) использование имени функции. Например

$\boldsymbol{f[\text{x\_}, \text{y\_}] = x + y^2}$

$2 \sim f \sim 3$

11

или

$1 \sim f \sim 2 \sim f \sim 3$
14

3.2.2 Вычисление производных и дифференциалов

Любую стандартную функцию или функцию пользователя, созданную одним из приведенных выше способов можно дифференцировать. Для вычисления производны в символьной форме используются функции `Derivative` и `D`.

Функция `Derivative[n][f]` вычисляет n–ю производную функции `f` одного аргумента. Например

Derivative[**1**][$\boldsymbol{f}$]
f'

Вместо использования полного имени функции `Derivative` можно писать обычное обозначение со штрихом. Так записи f', f'' эквивалентны Derivative[1][f] и Derivative[2][f]. Например

$\boldsymbol{y}$[**x_**] = $\boldsymbol{x^3}$;
$\boldsymbol{y'}[\boldsymbol{x}]$
$\boldsymbol{y''}[\boldsymbol{x}]$
$3x^2$
$6x$

Функция `Derivative` применяется к функции (не к выражению) и возвращает функцию. Например

$\boldsymbol{f}$[**x_**]:= $\boldsymbol{x^2}$**Sin**[$\boldsymbol{x}$]
$\boldsymbol{f'}$
$2\text{Sin}[\#1]\#1 + \text{Cos}[\#1]\#1^2\&$

Результат команды f' является функцией. Поэтому допустима запись

$\boldsymbol{f'}[\boldsymbol{x}]$
$x^2\text{Cos}[x] + 2x\text{Sin}[x]$

и

$\boldsymbol{f'}[\boldsymbol{\pi/2}]$
π

При использовании штрихов имеются некоторые особенности, вытекающие из операторной природы функции `Derivative`. Следующая форма команды дифференцирования не работает.

$(\boldsymbol{x^2} + \textbf{Sin}[\boldsymbol{x}])'$
$(x^2 + \text{Sin}[x])'$

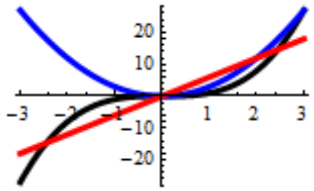
Однако работает

$((\#^{\boldsymbol{2}} + \textbf{Sin}[\#])\&)'[\boldsymbol{x}]$
$2x + \text{Cos}[x]$

Штрих должен стоять после функции как в последнем примере или после имени функции как в примерах до этого, а не после выражения! Поэтому можно писать

Cos′
$-\text{Sin}[\#1]\&$

$\mathbf{Cos'[x]}$
$-\mathrm{Sin}[x]$
Функция `Derivative` удобна при построения графиков производных.
$\mathbf{Plot[\{y[x], y'[x], y''[x]\}, \{x, -3, 3\}, PlotStyle \to \{Black, Blue, Red\}]}$



Общая форма функции Derivative[n1, n2, ...][f] вычисляет производную f по первой переменной n_1 раз, n_2 раз по второй переменной и т.д.
$\mathbf{g[x\_, y\_] = Sin[xy];}$
$\mathbf{Derivative[2, 3][g][x, y]}$
$-6x\mathrm{Cos}[xy] + x^3y^2\mathrm{Cos}[xy] + 6x^2y\mathrm{Sin}[xy]$
Можно использовать запись Derivative[-n][f], которая представляет неопределенный интеграл n-го порядка
$\mathbf{y[x\_] = x^3;}$
$\mathbf{Derivative[-1][y]}$
$\frac{\#1^4}{4}\&$
$\mathbf{Derivative[-1][y][x]}$
$\frac{x^4}{4}$
$\mathbf{Derivative[-2][y][x]}$
$\frac{x^5}{20}$
Функция `D` используется для дифференцирования выражений. В формате `D[f,x]` она вычисляют частную производную выражения f по переменной x.
$\mathbf{D[x^n, x]}$
$n\,x^{-1+n}$
В формате `D[f,{x,n}]` вычисляется частная производная n-го порядка выражения f по переменной x.
$\mathbf{D[x^n, \{x, 4\}]}$
$(-3+n)(-2+n)(-1+n)nx^{-4+n}$
В формате D[f,x_1,x_2,…] вычисляется смешанная производная $\frac{\partial^n f(x_1, x_2, \ldots)}{\partial x_1 \partial x_2 \ldots}$. При этом дифференцирование выполняется сначала по x_1, потом по x_2 и т.д.
$\mathbf{D[(x^2 + y^2)^3, x, y]}$
$24\,x\,y\,(x^2 + y^2)$
$\mathbf{f[x\_, y\_] := Sin[xy];}$
$\mathbf{D[f[x, y], x, \{y, 2\}]}$
$-x^2y\mathrm{Cos}[xy] - 2x\mathrm{Sin}[xy]$

Команда `D[f,{{x1,x2,…}}]` вычисляет вектор производных $\left\{\frac{\partial f}{\partial x_1},\frac{\partial f}{\partial x_2},\ldots\right\}$ скалярной функции `f`.

$\boldsymbol{D[f[x,y],\{\{x,y\}\}]}$

$\{f^{(1,0)}[x,y],f^{(0,1)}[x,y]\}$

$\boldsymbol{D[(x^2+y^2)^2,\{\{x,y\}\}]}$

$\{4x(x^2+y^2),4y(x^2+y^2)\}$

Запись `D[f,x]` может быть заменена на $\partial_x f$. Для ввода нужно набрать `Esc-pd-Esc`, затем `Ctrl-_` (подчеркивание), затем ввести *x*.

$\boldsymbol{\partial_x \mathbf{Sin}[x]}$

$\mathrm{Cos}[x]$

$\boldsymbol{\partial_x (a\,x^2)}$

$2a\,x$

Запись `D[f,x,y]` может быть записана в виде $\partial_{x,y} f$. Вводить надо также как предыдущий символ $\partial_x f$. При этом запятую нужно вводить, но она может быть невидимой. Для ввода невидимой запятой нужно набрать комбинацию клавиш `Esc-,(запятая)-Esc`.

$\boldsymbol{\partial_{x\,y}(x^2+y^2)^3}$ (* невидимая запятая *)

$24\,x\,y\,(x^2+y^2)$

$\boldsymbol{\partial_{x,x,y}\mathbf{Sin}[xy]}$ (* видимые запятые *)

$-xy^2\mathrm{Cos}[xy]-2y\mathrm{Sin}[xy]$

Построим для примера таблицу производных функции

$\boldsymbol{f[\mathrm{x\_}]=x^6+\mathbf{Sin}[x]}$;

$\boldsymbol{\mathbf{Grid}[\mathbf{Table}[\{\mathbf{ToString}[i]<>"-я",\partial_{\{x,i\}}f[x]\},\{i,8\}],\mathbf{Frame}\to\mathbf{All}]}$

1-я	6 x^5 + Cos[x]
2-я	30 x^4 - Sin[x]
3-я	120 x^3 - Cos[x]
4-я	360 x^2 + Sin[x]
5-я	720 x + Cos[x]
6-я	720 - Sin[x]
7-я	-Cos[x]
8-я	Sin[x]

Вы можете дифференцировать неопределенные функции. Вот, например, правило дифференцирования сложной функции

$\boldsymbol{\partial_x f[g[x]]}$

$f'[g[x]]g'[x]$

Вот формула дифференцирования произведения трех функций

$\boldsymbol{\partial_x(u[x]v[x]w[x])}$

$v[x]w[x]u'[x]+u[x]w[x]v'[x]+u[x]v[x]w'[x]$

Если какая – либо переменная неявно зависит от переменной дифференцирования, то это можно указать с помощью опции `NonConstants`. Например,

$\boldsymbol{D[x^3 + y^3, x, \text{NonConstants} \to y]}$

$3x^2 + 3y^2 D[y, x, \text{NonConstants} \to \{y\}]$

Эту опцию можно использовать, чтобы имитировать дифференцирование функции, заданной неявно

$\boldsymbol{\text{dd} = D[x^3 + y^3 == 1, x, \text{NonConstants} \to y]}$

$3x^2 + 3y^2 D[y, x, \text{NonConstants} \to \{y\}] == 0$

$\boldsymbol{\text{dp} = \text{dd}/.\{D[y, x, \text{NonConstants} \to \{y\}] \to y'\}}$

$3x^2 + 3y^2 y' == 0$

$\boldsymbol{\text{Solve}[\text{dp}, y']}$

$\{\{y' \to -\frac{x^2}{y^2}\}\}$

Функция `Dt[f]` вычисляет полный дифференциал выражения f.

$\boldsymbol{\text{Dt}[x^2 y^3]}$

$2xy^3 \text{Dt}[x] + 3x^2 y^2 \text{Dt}[y]$

Функция `Dt[f,x]` вычисляет полную производную df/dx выражения f, полагая, что другие параметры могут зависеть от переменной x.

$\boldsymbol{\text{Dt}[x^n, x]}$

$x^n \left(\frac{n}{x} + \text{Dt}[n, x]\text{Log}[x]\right)$

$\boldsymbol{\text{Dt}[\text{Sin}[x\, y], x]}$

$\text{Cos}[xy](y + x\text{Dt}[y, x])$

Можно сказать, что команда `D[f,x]` аналогична записи $\partial f/\partial x$, а команда `Dt[f,x]` аналогична записи df/dx.

Вторая полная производная выглядит несколько сложнее.

$\boldsymbol{\text{Dt}[x^2 y^3, \{x, 2\}]}$

$2y^3 + 12xy^2 \text{Dt}[y, x] + x^2(6y\text{Dt}[y, x]^2 + 3y^2 \text{Dt}[y, \{x, 2\}])$

Если в определении функции содержатся какие – либо константы, то их производные равняются нулю. Функции `Dt` это можно указать с помощью опции `Constants`, в значении которой перечислить имена этих констант. Например,

$\boldsymbol{\text{Dt}[a\, x\, y + \text{Sin}[k\, x], x, \text{Constants} \to \{a, k\}]}$

$a\, y + k\, \text{Cos}[k\, x] + a\, x\, \text{Dt}[y, x, \text{Constants} \to \{a, k\}]$

Полная производная функции одной переменной совпадает с ее частной производной

$\boldsymbol{\text{Dt}[\text{Sin}[x], x] == D[\text{Sin}[x], x]}$

`True`

Используя приведенные функции, можно решать классические задачи математического анализа. Например, можно построить частичную сумму ряда Тейлора

$\boldsymbol{\text{kf} = \text{Table}[D[\text{Log}[1 + x], \{x, n\}]/n!, \{n, 0, 8\}]/.\{x \to 0\}}$

$\{0, 1, -\frac{1}{2}, \frac{1}{3}, -\frac{1}{4}, \frac{1}{5}, -\frac{1}{6}, \frac{1}{7}, -\frac{1}{8}\}$

$\mathbf{Sum[kf[[i]]x^{i-1}, \{i, 1, 8\}]}$

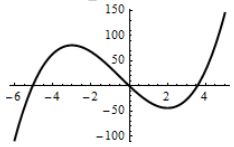
$$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \frac{x^7}{7}$$

Можно находить точки экстремума

$\mathbf{Clear[f]}$

$\mathbf{f[x\_] = 2\,x^3 + 3x^2 - 36x}$

$\mathbf{Plot[f[x], \{x, -6, 5\}, PlotRange \to All]}$



$\mathbf{Solve[D[f[x], x] == 0, x]}$

$\{\{x \to -3\}, \{x \to 2\}\}$

И точки перегиба

$\mathbf{Solve[D[f[x], \{x, 2\}] == 0, x]}$

$$\{\{x \to -\frac{1}{2}\}\}$$

Использование производных иногда полезно для доказательства функциональных тождеств. Хорошо известен следующий факт. Если две функции $f(x)$ и $g(x)$ определены и непрерывны в промежутке $\aleph$ и внутри него имеют конечные производные $f'(x), g'(x)$, причем $f'(x) = g'(x)$ внутри $\aleph$, то эти функции во всем промежутке $\aleph$ разнятся лишь на постоянную: $f(x) = g(x) + C \ \ (C = const)$.

Пример. Рассмотрим две функции *arctg x* и $\arcsin\frac{x}{\sqrt{1+x^2}} \quad (-\infty < x < \infty)$.

Вычислим их производные.

$\mathbf{f1[x\_] = ArcTan[x];}$

$\mathbf{f2[x\_] = ArcSin[\frac{x}{\sqrt{1 + x^2}}];}$

$\mathbf{f1'[x]}$

$$\frac{1}{1 + x^2}$$

$\mathbf{Simplify[f2'[x], Assumptions :\to x \in Reals]}$

$$\frac{1}{1 + x^2}$$

Производные совпадают во всем промежутке $-\infty < x < \infty$ и, следовательно, эти функции во сем промежутке разнятся на постоянную, которую можно определить в произвольной точке, например, при $x = 0$. Имеем

$\mathbf{Const = f1[0] - f2[0]}$

0

Т.о. $arctg\ x = \arcsin\frac{x}{\sqrt{1+x^2}} \quad (-\infty < x < \infty)$.

□

Для численного определения значения производной в точке предназначена функция ND. Она находится в пакете расширений `NumericalCalculus` (в *Mathematica* 5 в пакете `<<NumericalMath`NLimit`).

```
Needs["NumericalCalculus`"]
```

В формате `ND[expr,x,x0]` функция выполняет численную апроксимацию первой производной по x в точке x_0.

```
ND[Exp[Sin[x]], x, 2]
```

```
-1.03312
```

В формате `ND[expr,{x,n},x0]` функция выполняет численную апроксимацию n – ой производной в точке x_0.

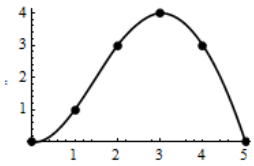
```
ND[xSin[x], {x, 2}, 0]
```

```
2.
```

`ND` полезна при вычислении производных функций, заданных численно.

```
points = {{0, 0}, {1, 1}, {2, 3}, {3, 4}, {4, 3}, {5, 0}};
ifun = Interpolation[points]
ifun[2. 5]
Plot[ifun[x], {x, 0, 5}, Epilog → {PointSize[0. 04], Map[Point, points]}]
```

```
InterpolatingFunction[{{0,5}},"<>"]
3.6875
```



Функция `Interpolation` строит интерполяционную функцию, определяемую списком **points**. Используя функцию `ND`, можно вычислять значение производной в конкретных точках.

```
ND[ifun[x], x, 1]
```

```
1.83333
```

Однако, объект InterpolatingFunction, созданный функцией `Interpolation` является «настоящей» функцией и его можно дифференцировать символьно.

```
D[ifun[x], x]/. x → 1.
```

```
1.83333
```

Производную в точке можно было бы вычислить по – другому так

```
ifun'[1.]
```

```
1.83333
```

Функция `ND`, в отличие от `D`, умеет вычислять односторонние производные. Подробнее с ней вы можете познакомиться по справочной системе.

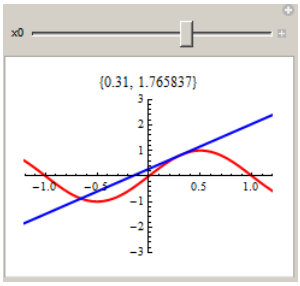
3.2.3 Геометрические приложения производных

Из курсов математического анализа известно, что значение производной в точке рано тангенсу наклона касательной к графику функции в этой точке. Уравнение касательной к кривой, заданной явно $y = f(x)$, в точке x_0 имеет вид

$$y = f(x_0) + f'(x_0)(x - x_0)$$

Пример. Касательная к графику функции $y = \sin \pi x$.

```
DynamicModule[{f, f1},
  f[x_] := Sin[πx];
  f1[x_, x0_] := f[x0] + f'[x0](x - x0);
  Manipulate[
    Plot[{f[x], f1[x, x0]}, {x, -1.5, 1.5},
         PlotRange → {{-1.2, 1.2}, {-3, 3}},
         PlotLabel → {x0, f'[x0]},
         PlotStyle → {{Red, Thickness[0.01]}, {Blue, Thickness[0.01]}}],
  {{x0, 0}, -1, 1}]]
```



Перемещая ползунок, вы будете менять точку касания x_0 и, соответственно будет перерисовываться касательная, а в заголовке графика будет отображаться текущее значение x_0 точки касания и значение производной в этой точке.

□

Пусть кривая задана неявным уравнением $F(x, y) = 0$ и точка (x_0, y_0) принадлежит кривой. Уравнение касательной к кривой в этой точке имеет вид

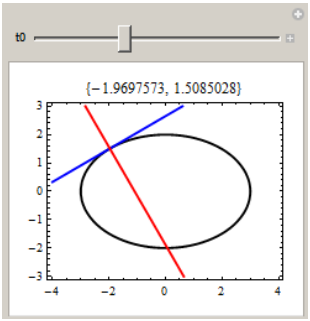
$$F'_x(x_0, y_0)(x - x_0) + F'_y(x_0, y_0)(y - y_0) = 0,$$

а уравнение нормали

$$F'_x(x_0, y_0)(y - y_0) - F'_y(x_0, y_0)(x - x_0) = 0$$

Пример. Касательная и нормаль к эллипсу, заданному неявным уравнение

```
DynamicModule[{a = 3, b = 2, t0, x0, y0, eq1, eq2, eq3, pc1, pc2},
  eq1[x_, y_] = x^2/a^2 + y^2/b^2 - 1;
  eq2[x0_, y0_] = Derivative[1, 0][eq1][x0, y0](x - x0) +
                  Derivative[0, 1][eq1][x0, y0](y - y0);
  eq3[x0_, y0_] = Derivative[1, 0][eq1][x0, y0](y - y0) -
                  Derivative[0, 1][eq1][x0, y0](x - x0);
  pc1 = ContourPlot[eq1[x, y] == 0,
                  {x, -a - 1, a + 1}, {y, -b - 1, b + 1},
                  ContourStyle → {Black, Thickness[0.01]}];
  Manipulate[ x0 = aCos[t0]; y0 = bSin[t0];
    pc2 = ContourPlot[{eq2[x0, y0] == 0, eq3[x0, y0] == 0},
                      {x, -a - 1, a + 1}, {y, -b - 1, b + 1},
                       ContourStyle → {{Blue, Thickness[0.01]},
                                       {Red, Thickness[0.01]}}];
    Show[pc1, pc2, PlotLabel → {x0, y0}, AspectRatio → Automatic],
  {{t0, π/4}, 0, 2π}]]
```



Положение ползунка определяет точку на эллипсе, в которой строятся касательная и нормаль. В заголовке графика отображаются текущие координаты точки.

□

Единичный касательный вектор к плоской кривой, заданной параметрически $x = x(t)$, $y = y(t)$, в точке $t = t_0$ равен

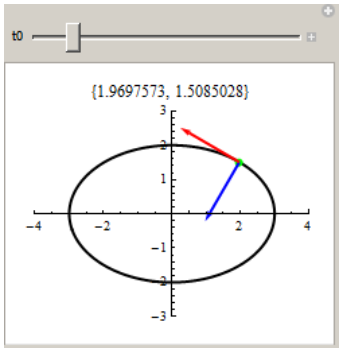
$$\mathbf{T} = (T_x, T_y) = \left(\frac{x'(t_0)}{\sqrt{(x'(t_0))^2 + (y'(t_0))^2}}, \frac{y'(t_0)}{\sqrt{(x'(t_0))^2 + (y'(t_0))^2}} \right).$$

Тогда касательная к кривой в точке $t = t_0$ может быть представлена уравнениями $x(t) = x(t_0) + T_x \cdot t$; $y(t) = y(t_0) + T_y \cdot t$.

Единичный вектор нормали к кривой в точке $t = t_0$ равен $\mathbf{N} = (-T_y, T_x)$. Тогда уравнение нормали в этой точке можно записать в виде $x(t) = x(t_0) - T_y \cdot t$; $y(t) = x(t_0) + T_x \cdot t$.

Пример. Касательная и нормаль к эллипсу, заданному параметрически.

```
DynamicModule[{a = 3, b = 2, t0, tx, ty, x, y, xt, yt, xn, yn, pe, pa, pp},
  {x[t_], y[t_]} = {aCos[t], bSin[t]};   (* радиус-вектор *)
  {tx[t_], ty[t_]} = {x'[t], y'[t]} / Sqrt[x'[t]^2 + y'[t]^2];   (* единичный вектор касательной *)
  xt[t_, t0_] = x[t0] + tx[t0]t;   (* уравнение касательной *)
  yt[t_, t0_] = y[t0] + ty[t0]t;
  xn[t_, t0_] = x[t0] - ty[t0]t;   (* уравнение нормали *)
  yn[t_, t0_] = y[t0] + tx[t0]t;
  Manipulate[
    pe = ParametricPlot[{x[t], y[t]}, {t, 0, 2π}];
    pp = Graphics[{Green, PointSize[0.03], Point[{x[t0], y[t0]}]}];
    pa = Graphics[{
      {Blue, Thickness[0.01], Arrow[{{x[t0], y[t0]}, {xn[2, t0], yn[2, t0]}}]},
      {Red, Thickness[0.01], Arrow[{{x[t0], y[t0]}, {xt[2, t0], yt[2, t0]}}]}}];
    Show[pe, pp, pa, PlotRange → {{-4, 4}, {-3, 3}},
                     PlotLabel → {x[t0], y[t0]}],
  {t0, 0, 2π}]]
```



Перемещая ползунок, вы будете менять точку кривой и, соответственно будут перерисовываться вектор касательной и нормали, а в заголовке графика будут отображаться текущие значения (x_0, y_0) координат точки.

□

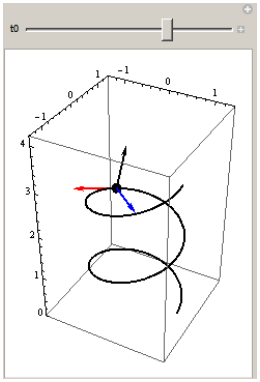
Единичный касательный вектор к пространственной кривой, заданной параметрически $\mathbf{r}(t)=(x(t),y(t),z(t))$, в точке $t=t_0$ равен $\mathbf{T}=(T_x,T_y,T_z)=\dfrac{\mathbf{r}'(t_0)}{\|\mathbf{r}'(t_0)\|}$.

Единичный вектор главной нормали равен $\mathbf{N}=(N_x,N_y,N_z)=\dfrac{\mathbf{r}''(t_0)}{\|\mathbf{r}''(t_0)\|}$.

Единичный вектор бинормали равен $\mathbf{B}=(B_x,B_y,B_z)=[\mathbf{T},\mathbf{N}]$. Тогда векторные уравнения касательной, главной нормали и бинормали будут иметь вид: $\mathbf{r}_t(t)=\mathbf{r}(t_0)+\mathbf{T}\cdot t$, $\mathbf{r}_n(t)=\mathbf{r}(t_0)+\mathbf{N}\cdot t$, $\mathbf{r}_b(t)=\mathbf{r}(t_0)+\mathbf{B}\cdot t$.

Пример. Репер Френе.

```
DynamicModule[{a = 1, b = 1, m = 4, tmax = 4π, db = 0.4,
                t0, r, tr, nr, br, rt, rn, rb, pc, pa, pp},
  r[t_] = {aCos[t], bSin[t], t/m};  (* радиус-вектор *)
  tr[t_] = Normalize[r'[t]];    (* единичный вектор касательной *)
  nr[t_] = Normalize[r''[t]];   (* единичный вектор главной нормали *)
  br[t_] = Simplify[Cross[tr[t], nr[t]]];  (* единичный вектор бинормали *)
  rt[t_, t0_] = r[t0] + tr[t0]t;   (* уравнение касательной *)
  rn[t_, t0_] = r[t0] + nr[t0]t;   (* уравнение главной нормали *)
  rb[t_, t0_] = r[t0] + br[t0]t;   (* уравнение бинормали *)
  Manipulate[
    pc = ParametricPlot3D[r[t], {t, 0, tmax}];
    pp = Graphics3D[{PointSize[0.03], Point[r[t0]]}];
    pa = Graphics3D[{
          {Blue, Arrow[{r[t0], rn[1, t0]}]},
          {Red, Arrow[{r[t0], rt[1, t0]}]},
          {Black, Arrow[{r[t0], rb[1, t0]}]}}];
    Show[pc, pp, pa, BoxRatios → {2a, 2b, tmax/m},
          PlotRange → {{-a - db, a + db}, {-b - db, b + db}, {0, 4.2}}],
  {{t0, π}, 0, tmax}]]
```



Перемещая ползунок, вы будете менять точку кривой и соответственно будут перерисовываться вектора репера Френе.

Обратите внимание, что в приведенном коде вектор – функции **r[t]**, **tr[t]** и другие являются списками из трех скалярных функций. Функция `Normalize` преобразует вектор в вектор единичной длины.

□

Для поверхности, заданной неявным уравнением $F(x,y,z)=0$, уравнение касательной плоскости в точке (x_0,y_0,z_0) имеет вид

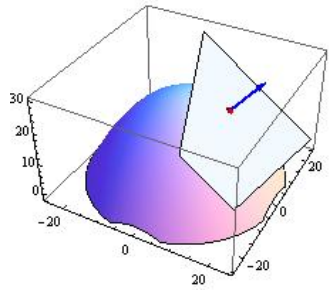
$$F'_x(x_0,y_0,z_0)(x-x_0)+F'_y(x_0,y_0,z_0)(y-y_0)+F'_z(x_0,y_0,z_0)(z-z_0)=0$$

Уравнение нормали к поверхности в этой точке

$$\frac{x-x_0}{F'_x(x_0,y_0,z_0)}=\frac{y-y_0}{F'_y(x_0,y_0,z_0)}=\frac{z-z_0}{F'_z(x_0,y_0,z_0)}$$

Пример. Касательная плоскость и вектор нормали к эллиптическому параболоиду

```
eq1 =.; a = 5; b = 5;
eq1[x_, y_, z_] = x^2/a^2 + y^2/b^2 + z - 25;
x0 = 10; y0 = 4; z0 = 25 - x0^2/a^2 - y0^2/b^2;
FX = Derivative[1, 0, 0][eq1][x0, y0, z0];
FY = Derivative[0, 1, 0][eq1][x0, y0, z0];
FZ = Derivative[0, 0, 1][eq1][x0, y0, z0];
eq2 = FX(x - x0) + FY(y - y0) + FZ(z - z0);
n[t_] = {x0, y0, z0} + t {FX, FY, FZ};
pa = Graphics3D[{Blue, Thickness[0.01], Arrow[{n[0], n[10]}]}];
ps = ContourPlot3D[eq1[x, y, z] == 0,
        {x, -5a, 5a}, {y, -5b, 5b}, {z, -2, 28}, Mesh → None];
pp = ContourPlot3D[eq2 == 0,
        {x, -5a, 5a}, {y, -5b, 5b}, {z, -2, 28}, Mesh → None];
pt = Graphics3D[{Red, Sphere[{x0, y0, z0}, 1]}];
Show[ps, pp, pt, pa, PlotRange → All, BoxRatios → {50, 50, 30}]
```



□

Можно построить касательные плоскости и нормали к поверхностям, заданным явно $z = f(x, y)$ или заданным параметрически $x = x(u,v),\ y = y(u,v),\ z = z(u,z)$. Предоставляем читателю возможность самостоятельно рассмотреть эти примеры.

3.3 Интегрирование

3.3.1 Символьное вычисление интегралов

Для вычисления неопределенного интеграла $\int f(x)dx$ используется функция `Integrate[f,x]`.

$$\mathbf{Integrate[Cos[}x\mathbf{]^2}, x\mathbf{]}$$

$$\frac{x}{2} + \frac{1}{4}\text{Sin}[2x]$$

Обратите внимание, что константа интегрирования не добавляется.

Вместо имени функции можно использовать классическое обозначение ∫.

$$\int x^2 \, dx$$

$$\frac{x^3}{3}$$

Трафарет ввода знака интеграла можно найти на панели специальных символов «Basic Math Input». Можно использовать также специальные комбинации клавиш. Для ввода «крючка» интеграла наберите `Esc-int-Esc`, затем введите подынтегральное выражение, затем специальный значок дифференциала `Esc-dd-Esc` и имя переменной интегрирования.

$$\int \mathbf{Sqrt[}x + \mathbf{Sqrt[}x\mathbf{]]}\, dx$$

$$\frac{1}{12}\sqrt{\sqrt{x} + x}(-3 + 2\sqrt{x} + 8x) + \frac{1}{8}\text{Log}[1 + 2\sqrt{x} + 2\sqrt{\sqrt{x} + x}]$$

Можно интегрировать выражения, содержащие более чем одну переменную; другие переменные будут трактоваться, как константы

$$\int a\, x^2 \, dx$$

$$\frac{a\, x^3}{3}$$

Можно вычислять интеграл от интеграла

$$\mathbf{Integrate[}x^2 y^3, x, y\mathbf{]}$$

$$\frac{x^3 y^4}{12}$$

или по – другому

$$\int\int x^2 y^3 \, dy \, dx$$

$$\frac{x^3 y^4}{12}$$

Mathematica умеет интегрировать рациональные, тригонометрические и много других функций

$$\int \frac{x^5}{x^3 + 3x^2 + 7x} \, dx$$

$$2x - \frac{3x^2}{2} + \frac{x^3}{3} - \frac{73\text{ArcTan}[\frac{3+2x}{\sqrt{19}}]}{\sqrt{19}} + \frac{15}{2}\text{Log}[7 + 3x + x^2]$$

Она «знает» много интегралов, которые могут быть определены в терминах специальных функций

$$\int \text{Exp}[-x^2] \, dx$$

$$\frac{1}{2}\sqrt{\pi}\text{Erf}[x]$$

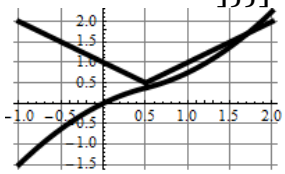
$$\int \frac{\text{Sin}[x^2]}{x^5} \, dx$$

$$-\frac{\text{Cos}[x^2]}{4x^2} - \frac{\text{Sin}[x^2]}{4x^4} - \frac{\text{SinIntegral}[x^2]}{4}$$

Mathematica умеет интегрировать кусочные функции

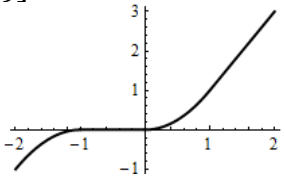
```
Grid[{{
  px = Integrate[Max[x, 1 - x], x],
  Plot[{Max[x, 1 - x], px}, {x, -1, 2},
       PlotStyle → {{Black, Thickness[0.02]}},
       GridLines → Automatic]}}]
```

$$\begin{cases} x - \frac{x^2}{2} & x \le \frac{1}{2} \\ \frac{1}{4} + \frac{x^2}{2} & \text{True} \end{cases}$$



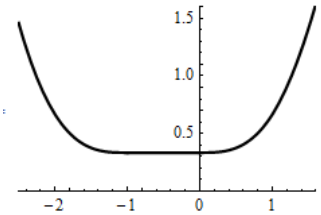

```
Grid[{{
  y[x_] = Piecewise[{{-(x + 1)^2, x < -1}, {0, x < 0}, {x^2, x < 1}}, 2x - 1],
  Plot[y[x], {x, -2, 2}]}}]
```

$$\begin{cases} -(1+x)^2 & x < -1 \\ 0 & x < 0 \\ x^2 & x < 1 \\ -1 + 2x & \text{True} \end{cases}$$




```
Grid[{{
  z = ∫ y[x] dx,
  Plot[z, {x, -2.5, 1.6}, PlotRange → {{-2.5, 1.6}, {0, 1.6}}]}}]
```

$$\begin{cases} -x - x^2 - \frac{x^3}{3} & x \le -1 \\ \frac{1}{3} & -1 < x \le 0 \\ \frac{1}{3} + \frac{x^3}{3} & 0 < x \le 1 \\ \frac{2}{3} - x + x^2 & \text{True} \end{cases}$$



Проверить правильность вычисления интегралов можно дифференцированием

$$\mathbf{i1} = \int \int \frac{\mathbf{1}}{\boldsymbol{x}^{\mathbf{2}} + \mathbf{4}} \, \mathbb{d}\boldsymbol{x} \, \mathbb{d}\boldsymbol{x}$$

$$\frac{1}{2}\left(x\text{ArcTan}\left[\frac{x}{2}\right] - \text{Log}[4 + x^2]\right)$$

$$\mathbf{FullSimplify}[\boldsymbol{D}[\mathbf{i1}, \{\boldsymbol{x}, \mathbf{2}\}]]$$

$$\frac{1}{4 + x^2}$$

Напомним, что неопределенный интеграл от функции можно вычислить, используя функцию `Derivative`.

$$\mathbf{Derivative}[-\mathbf{1}]\left[\frac{\#^{\mathbf{2}}}{\#^{\mathbf{3}} + \mathbf{1}}\&\right][\boldsymbol{x}]$$

$$\frac{1}{3}\text{Log}[1 + x^3]$$

Когда *Mathematica* не может взять интеграл, она возвращает команду обратно

$$\mathbf{Integrate}\left[\frac{\mathbf{1}}{\boldsymbol{x} + \mathbf{Cos}[\boldsymbol{x}]}, \boldsymbol{x}\right]$$

$$\int \frac{1}{x + \text{Cos}[x]} \, \mathbb{d}x$$

Иногда для вычисления неопределенного интеграла следует учитывать условия, накладываемые на независимую переменную, поскольку без дополнительных условий интеграл не вычисляется. Например, условия важны при интегрировании кусочных функций

$$\mathbf{Integrate}[\mathbf{Abs}[\boldsymbol{x}], \boldsymbol{x}]$$

$$\int \text{Abs}[x] \, \mathbb{d}x$$

$$\mathbf{Integrate}[\mathbf{Abs}[\boldsymbol{x}], \boldsymbol{x}, \mathbf{Assumptions} \text{->} \boldsymbol{x} \in \mathbf{Reals}]$$

$$\begin{cases} -\frac{x^2}{2} & x \le 0 \\ \frac{x^2}{2} & \text{True} \end{cases}$$

Заметим, что разные формы одного и того же неопределенного интеграла могут давать ответы, отличающиеся константой. Например

$$\mathbf{i1} = \mathbf{Integrate}[\mathbf{1} + (\boldsymbol{x} + \mathbf{1})\text{^}\mathbf{3}, \boldsymbol{x}]$$

$$x + \frac{1}{4}(1 + x)^4$$

$$\mathbf{i2} = \mathbf{Integrate}[\mathbf{Expand}[\mathbf{1} + (\boldsymbol{x} + \mathbf{1})\text{^}\mathbf{3}], \boldsymbol{x}]$$

$$2x + \frac{3x^2}{2} + x^3 + \frac{x^4}{4}$$

```
Simplify[i1 − i2]
```

$$\frac{1}{4}$$

Для вычисления определенного интеграла используется второй аргумент в виде списка `Integrate[f,{x,xmin,xmax}]`. Следующая команда интегрирует x^2 от 0 до 1.

```
Integrate[x^2,{x,0,1}]
```

$$\frac{1}{3}$$

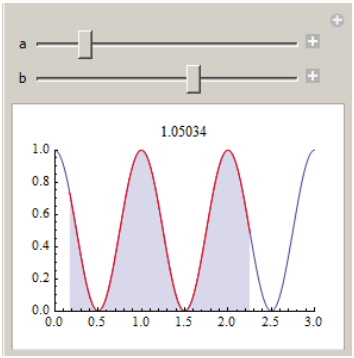
или

$$\int_0^1 x^2 \, dx$$

Трафарет ввода определенного интеграла можно найти на панели специальных символов «Basic Math Input». Можно также использовать специальные комбинации клавиш. Для ввода «крючка» интеграла наберите `Esc-int-Esc`. Потом надо ввести подстрочный и надстрочный индексы у значка интеграла $\int$. Для этого введите `Ctrl-_` (подчеркивание), затем `Ctrl-%` (или `Ctrl-^`, затем `Ctrl-%`). Введите значения нижнего и верхнего пределов интегрирования. `Ctrl-пробел` возвращает курсор на нормальный уровень ввода, где введите подынтегральное выражение, затем введите специальный значок дифференциала `Esc-dd-Esc` и имя переменной интегрирования.

Следующий код поясняет геометрический смысл определенного интеграла $\int_a^b f(x)dx$ как площади под кривой $y = f(x)$ на участке от a до b.

```
Manipulate[
  p1 = Plot[Cos[πx]^2,{x,0,3},PlotRange → {{0,3},{0,1}}];
  p2 = Plot[Cos[πx]^2,{x,a,b},Filling → Axis,PlotStyle → Red,
                              PlotRange → {{0,3},{0,1}}];
  s = Integrate[Cos[πx]^2,{x,a,b}];
  Show[p1,p2,PlotLabel → s],
{{a,0},0,1},{{b,2},1.1,3}]
```



Перемещая бегунки, связанные с параметрами a и b, вы можете наблюдать за областью, площадь которой вычисляется и отображается в заголовке.

Если под интегралом присутствует параметр, то результат будет от него зависеть и в ответе может присутствовать условие

$$\int_0^1 x^n \, dx$$

$$\text{ConditionalExpression}\left[\frac{1}{1+n}, \text{Re}[n] > -1\right]$$

Здесь `ConditionalExpression[expr,cond]` является символической конструкцией, которая представляет выражение `expr` только, если условие `cond` истинно. Вы можете использовать `ConditionalExpression` как функцию. Например,

$$y[\text{n\_}] = \int_0^1 x^n \, dx;$$

```
y[5]
```

$$\frac{1}{6}$$

При вычислении определенного интеграла условие можно учитывать сразу. Для этого используется опция `Assumptions` или функция `Assuming`.

$$\textbf{Integrate}[x\text{^}n, \{x, 0, 1\}, \textbf{Assumptions} -> n > 0]$$

$$\frac{1}{1+n}$$

или

$$\textbf{Assuming}[n > 0, \textbf{Integrate}[x\text{^}n, \{x, 0, 1\}]]$$

$$\frac{1}{1+n}$$

У функции `Assuming` два аргумента: первый – список предположений/условий, второй – обрабатываемое выражение.

Можно отключить «условный» вывод результата с помощью опции `GenerateConditions → False`.

$$\textbf{Integrate}[x^n, \{x, 0, 1\}, \textbf{GenerateConditions} \to \textbf{False}]$$

$$\frac{1}{1+n}$$

Mathematica может вычислять несобственные интегралы с бесконечными пределами интегрирования

$$\int_0^\infty \frac{1}{1+x^4} \, dx$$

$$\frac{\pi}{2\sqrt{2}}$$

$$\textbf{Integrate}[\textbf{Exp}[-cx^2], \{x, -\infty, \infty\}, \textbf{Assumptions} \to \textbf{Re}[c] > 0]$$

$$\frac{\sqrt{\pi}}{\sqrt{c}}$$

При разных условиях/предположениях интеграл может принимать разные значения

Table[

$$\textbf{Integrate}\left[\frac{\textbf{Cos}[x](1-\textbf{Cos}[ax])}{x^2},\{x,0,\infty\},\textbf{Assumptions}:\to \textbf{as}\right],$$

$\{\textbf{as},\{a>1,-1<a<1,a<-1\}\}]$

$\{\frac{1}{2}(-1+a)\pi,0,-\frac{1}{2}(1+a)\pi\}$

Можно вычислять несобственные интегралы на конечных отрезках.

$\textbf{Integrate}[\frac{1}{\sqrt{x}},\{x,0,1\}]$

2

Можно вычислять главное значение определенного интеграла, используя опцию `PrincipalValue->True`. Например, следующий интеграл не существует

$\textbf{Integrate}[1/x,\{x,-1,3\}]$

```
Integrate::idiv: Integral of 1/x does not converge on {-1,3}.≫
```

$\int_{-1}^{3}\frac{1}{x}dx$

Но существует его главное значение.

$\textbf{Integrate}[1/x,\{x,-1,3\},\textbf{PrincipalValue}->\textbf{True}]$

Log[3]

Аналогично у следующего интеграла существует только главное значение

$\textbf{Integrate}[\frac{1}{\textbf{Sinh}[2x+1]},\{x,-1,1\},\textbf{PrincipalValue}\to\textbf{True}]$

$\text{ArcTanh}[\frac{e}{1+e^2}]$

Вы можете пожелать строить графики неопределенных интегралов или определенных с переменными пределами интегрирования. Но при этом следует учитывать некоторые особенности. Например, следующая команда не работает

$\textbf{Plot}[\textbf{Integrate}[\textbf{Cos}[x],x],\{x,-4,4\}]$

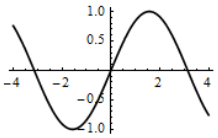
потому, что функция `Plot` имеет атрибут `HoldAll`.

$\textbf{Attributes}[\textbf{Plot}]$

{HoldAll, Protected, ReadProtected}

Он означает, что аргументы функции `Plot` не вычисляются символьно при выполнении команды. Чтобы вычислить выражения, предаваемые в качестве аргументов такой функции, следует использовать функцию `Evaluate`.

$\textbf{Plot}[\textbf{Evaluate}[\textbf{Integrate}[\textbf{Cos}[x],x]],\{x,-4,4\}]$



Следующая команда выполняется, но достаточно долго

$\mathbf{Plot}[\mathbf{Integrate}[\mathbf{Cos}[\boldsymbol{t}],\{\boldsymbol{t},-\boldsymbol{\pi},\boldsymbol{x}\}],\{\boldsymbol{x},-\boldsymbol{\pi},\boldsymbol{\pi}\}]$ (* не делайте так *)

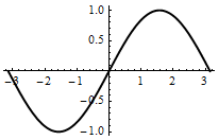
потому, что интеграл вычисляется символьно при каждом значении *x*. Чтобы интеграл вычислялся сразу (один раз) следует использовать функцию `Evaluate`.

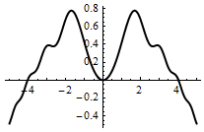
$\mathbf{Plot}[\mathbf{Evaluate}[\mathbf{Integrate}[\mathbf{Cos}[\boldsymbol{t}],\{\boldsymbol{t},-\boldsymbol{\pi},\boldsymbol{x}\}]],\{\boldsymbol{x},-\boldsymbol{\pi},\boldsymbol{\pi}\}]$ (* след. рис. слева *)

То же замечание касается такого примера (следующий рисунок справа).

$\mathbf{Plot}[\mathbf{Evaluate}[\int_{\mathbf{0}}^{\boldsymbol{x}}\int_{\mathbf{0}}^{\boldsymbol{h}}(\mathbf{Cos}[\boldsymbol{ht}]-\mathbf{0.1})\,\mathbb{d}\boldsymbol{t}\,\mathbb{d}\boldsymbol{h}],\{\boldsymbol{x},-\mathbf{5},\mathbf{5}\},\mathbf{Axes}\rightarrow\mathbf{True}]$

В этом примере без использования функции `Evaluate` на некоторых системах код выполняется так долго, что возможно вам придется прервать вычисления командой `Alt-.` (точка) или `Alt-,` (запятая).





Определенные интегралы, зависящие от параметра, можно дифференцировать по этому параметру. Например,

$\mathbf{Integrate}[\boldsymbol{f}[\boldsymbol{t}],\{\boldsymbol{t},\mathbf{0},\boldsymbol{x}\}]$

$$\int_0^x f[t]\,\mathbb{d}t$$

$\boldsymbol{D}[\%,\boldsymbol{x}]$

$$f[x]$$

Общая формула дифференцирования интеграла, зависящего от параметра, может быть получена следующей командой

$\mathbf{Clear}[\boldsymbol{F},\boldsymbol{f},\boldsymbol{g}]$

$\mathbf{Integrate}[\boldsymbol{F}[\boldsymbol{t},\boldsymbol{x}],\{\boldsymbol{t},\boldsymbol{g}[\boldsymbol{x}],\boldsymbol{f}[\boldsymbol{x}]\}]$

$$\int_{g[x]}^{f[x]} F[t,x]\,\mathbb{d}t$$

$\boldsymbol{D}[\%,\boldsymbol{x}]$

$$\int_{g[x]}^{f[x]} F^{(0,1)}[t,x]\,\mathbb{d}t + F[f[x],x]f'[x] - F[g[x],x]g'[x]$$

В одной команде можно комбинировать вычисление определенного и неопределенного интегралов.

$\mathbf{Integrate}[(\boldsymbol{x}+\boldsymbol{y})^{\mathbf{2}},\{\boldsymbol{y},\mathbf{0},\mathbf{1}\},\boldsymbol{x}]$

$$\frac{1}{6}x\,(2+3x+2x^2)$$

Для вычисления повторных интегралов добавьте больше аргументов в команду `Integrate`. Например, команда

```
Integrate[f,{x,xmin,xmax},{y,ymin,ymax}]
```

вычисляет повторный интеграл вида $\int_{x_{\min}}^{x_{\max}} d\,x \int_{y_{\min}}^{y_{\max}} f(x,y)\,d\,y$. Первым указывается интервал изменения той переменной, интегрирование по которой выполняется в последнюю очередь!

```
Integrate[x y, {x, 0, π}, {y, 0, x}]
```

$$\frac{\pi^4}{8}$$

или

$$\int_0^{\pi}\int_0^{x} x\,y\,dy\,dx$$

$$\frac{\pi^4}{8}$$

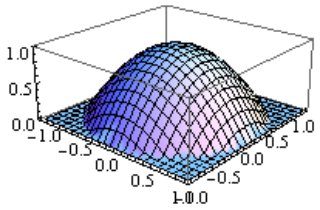
Функцию `Integrate` можно использовать для вычисления интегралов по области. Для этого можно использовать повторный интеграл предварительно «обнулив» функцию вне области. Вот пример интегрирования функции $f(x,y)=1-x^2-y^2$ по области единичного круга.

```
Integrate[If[x^2 + y^2 < 1, 1 - (x^2 + y^2), 0], {x, -1, 1}, {y, -1, 1}]
```

$$\frac{\pi}{2}$$

Ниже показан график «обрезанной» функции $f(x,y)$.

```
Plot3D[If[x^2 + y^2 < 1, 1 - (x^2 + y^2), 0], {x, -1, 1}, {y, -1, 1},
                                                    Mesh → {21, 21}]
```



Тот же интеграл можно вычислить по – другому

```
Integrate[(1 - (x^2 + y^2))Boole[x^2 + y^2 < 1], {x, -1, 1}, {y, -1, 1}]
```

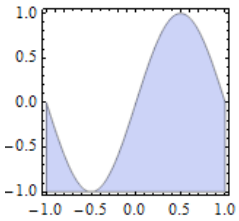
$$\frac{\pi}{2}$$

Функция `Boole[expr]` возвращает единицу, если `expr` принимает значение `True` и 0 – если `False`.

Область может задаваться логической комбинацией неравенств

```
RegionPlot[-1 < x < 1 && y < Sin[π x] && y > -1,
           {x, -1, 1}, {y, -1, 1}, AspectRatio → Automatic]
Integrate[(x + y)Boole[-1 < x < 1 && y < Sin[π x] && y > -1],
          {x, -1, 1}, {y, -1, 1}]
```

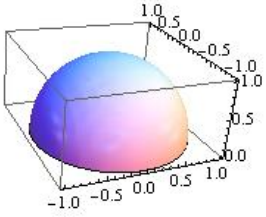
$$\frac{4-\pi}{2\pi}$$



Область может иметь любую размерность. В следующем примере мы вычисляем объем 3D области, ограниченной полусферой единичного радиуса.

```
Integrate[Boole[0 ≤ z ≤ 1 && x^2 + y^2 + z^2 ≤ 1],
    {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}]
RegionPlot3D[0 ≤ z ≤ 1 && x^2 + y^2 + z^2 ≤ 1,
    {x, -1, 1}, {y, -1, 1}, {z, 0, 1}, Mesh → None, BoxRatios → {1, 1, 0.5}]
```

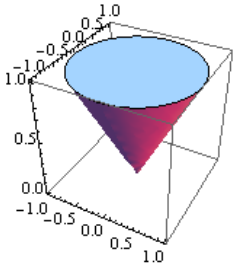
$$\frac{2\pi}{3}$$



Вот пример вычисления объема конуса.

```
Integrate[(x^2 + y^2)Boole[0 ≤ z ≤ 1 && x^2 + y^2 ≤ z^2],
        {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}]
RegionPlot3D[0 ≤ z ≤ 1 && x^2 + y^2 ≤ z^2,
        {x, -1, 1}, {y, -1, 1}, {z, 0, 1}, Mesh → None]
```

$$\frac{\pi}{10}$$

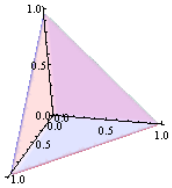


Пример. Вычислить интеграл

$$I = \iiint \frac{d\,x\,d\,y\,d\,z}{(1+x+y+z)^3}$$

распространенный на тетраэдр, ограниченный плоскостями $x=0,\ y=0,\ z=0$ и $x+y+z=1$. Вначале построим область интегрирования

```
RegionPlot3D[x > 0 && y > 0 && z > 0 && x + y + z ≤ 1,
        {x, 0, 1}, {y, 0, 1}, {z, 0, 1}, PlotPoints → 50, Mesh → None,
        PlotStyle → Opacity[0.5], Boxed → False, AxesOrigin → {0, 0, 0}]
```



Теперь вычисляем интеграл

```
Integrate[1/(1 + x + y + z)^3 Boole[x > 0 && y > 0 && z > 0 && x + y + z ≤ 1],
                                        {x, 0, 1}, {y, 0, 1}, {z, 0, 1}]
```

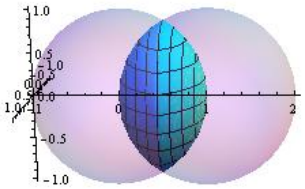
$$\frac{1}{16}(-5 + 8\text{Log}[2])$$

□

Пример. Найдем значение интеграла $I = \iiint y^2\, dx\, dy\, dz$ по области, являющейся общей частью двух шаров $x^2 + y^2 + z^2 \le R^2$ и $x^2 + y^2 + z^2 \le 2Ry$.

Вначале построим область интегрирования

```
R = 1;
opt = {PlotPoints → 50, PlotStyle → Opacity[0.3], Mesh → None,
       PlotRange → All};
sp1 = RegionPlot3D[x^2 + y^2 + z^2 ≤ R^2,
       {x, -R, R}, {y, -R, R}, {z, -R, R}, ##]& @@ opt;
sp2 = RegionPlot3D[x^2 + y^2 + z^2 ≤ 2Ry,
       {x, -R, R}, {y, 0, 2R}, {z, -R, R}, Evaluate[opt]];
sp3 = RegionPlot3D[x^2 + y^2 + z^2 ≤ R^2 && x^2 + y^2 + z^2 ≤ 2 R y,
       {x, -R, R}, {y, 0, R}, {z, -R, R}, PlotPoints → 50, PlotStyle → Cyan,
       MeshFunctions → {#2&, #3&}, Mesh → {7, 7, 7}];
Show[sp1, sp2, sp3, AxesOrigin → {0, -R, 0}, Boxed → False,
       BoxRatios → {2R, 3R, 2R}]
```



Обратите внимание на два способа применения списка опций **opt**. Теперь вычисляем интеграл

```
Integrate[y^2 Boole[x^2 + y^2 + z^2 ≤ R^2 && x^2 + y^2 + z^2 ≤ 2Ry],
       {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}, Assumptions :→ R > 0]
```

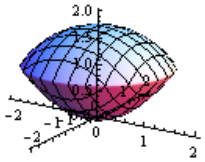
$$\frac{59\pi}{480}$$

□

Пример. Вычислим интеграл $I = \iiint\limits_V (x + y + z)^2\, dx\, dy\, dz$ по области V, являющейся общей частью параболоида $x^2 + y^2 \le 2az$ и шара $x^2 + y^2 + z^2 \le 3a^2$.

Вначале представим область интегрирования

```
a = 1;
RegionPlot3D[x^2 + y^2 ≤ 2 a z && x^2 + y^2 + z^2 ≤ 3a^2,
    {x, -2, 2}, {y, -2, 2}, {z, 0, 2}, PlotPoints → 100, Boxed → False,
    AxesOrigin → {0, 0, 0}, Mesh → {9, 9, 0}, BoxRatios → {4, 4, 2}]
```



Теперь вычисляем интеграл.

```
Integrate[(x + y + z)^2 Boole[x^2 + y^2 <= 2 a z && x^2 + y^2 + z^2 <= 3 a^2],
                {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}]
```

$$\frac{1}{30}(-97+108\sqrt{3})\pi$$

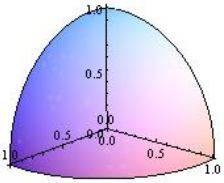
□

Пример. Вычислим интеграл

$$\iiint\limits_{\substack{x,y,z\geq 0\\ x^2+y^2+z^2\leq R^2}} \frac{x\,y\,z\,d\,x\,d\,y\,d\,z}{\sqrt{\alpha^2x^2+\beta^2y^2+\gamma^2z^2}} \quad (\alpha>\beta>\gamma>0)$$

Вначале представим область интегрирования

```
R = 1;
RegionPlot3D[x >= 0 && y >= 0 && z >= 0 && x^2 + y^2 + z^2 <= R^2,
{x, 0, 1}, {y, 0, 1}, {z, 0, 1}, PlotPoints → 50, Mesh → None,
AxesOrigin → {0, 0, 0}, Boxed → False]
```



Теперь вычисляем интеграл.

```
Clear[a, b, c, R];
Block[{$Assumptions = a > b && b > c && c > 0 && R > 0},
  Integrate[xyz/Sqrt[a^2 x^2 + b^2 y^2 + c^2 z^2] *
            Boole[x >= 0 && y >= 0 && z >= 0 && x^2 + y^2 + z^2 <= R^2],
  {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}]]
```

$$\frac{(ab+ac+bc)R^5}{15(a+b)(a+c)(b+c)}$$

Обратите внимание на способ наложения условий на параметры. Вместо использования опции `Assumptions` мы использовали переменную `$Assumptions`, область действия которой ограничили блоком.

□

Для вычисления объема $4-$х мерного шара единичного радиуса можно выполнить команду

```
Integrate[Boole[Sum[x[i]^2, {i, 4}] < 1],
                {x[1], -1, 1}, {x[2], -1, 1}, {x[3], -1, 1}, {x[4], -1, 1}]
```

$$\frac{\pi^2}{2}$$

При вычислении интегралов по области иногда также следует указывать условия/предположения, накладываемые на параметры. Например, при вычислении объема n – мерного шара радиуса r следует использовать условие/предположение относительно параметра $r > 0$.

```
n = 5;  (* размерность пространства *)
Integrate[Boole[Sum[x[i]^2, {i, n}] < r^2],
    Apply[Sequence, Table[{x[i], -∞, ∞}, {i, n}]], Assumptions → r > 0]
```

$$\frac{8\pi^2 r^5}{15}$$

Проверьте формулу для `n=2` и `n=3`.

Функцию `Integrate` можно применять к спискам выражений. В результате будет получен список интегралов.

```
Integrate[{x, x^2, x^3, x^4}, x]
```

$$\left\{\frac{x^2}{2}, \frac{x^3}{3}, \frac{x^4}{4}, \frac{x^5}{5}\right\}$$

```
Integrate[{x, x^2, x^3, x^4}, {x, 0, 1}]
```

$$\left\{\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}\right\}$$

```
Integrate[{{1, 2x, 3x^2}, {2x, 1, 4x^3}, {3x^2, 4x^3, 1}}, x]//MatrixForm
```

$$\begin{pmatrix} x & x^2 & x^3 \\ x^2 & x & x^4 \\ x^3 & x^4 & x \end{pmatrix}$$

Функция `Integrate` умеет вычислять интегралы от комплексных функций вещественного аргумента. В частности для функций вида $f(t) = u(t) + \boldsymbol{i}\, v(t)$ имеем $\int f(t)\,dt = \int u(t)\,dt + \boldsymbol{i}\int v(t)\,dt$. При этом нет необходимости в выделении вещественной и мнимой частей функции. Например,

```
f[t_] = (t - i)^3;
u[t_] = ComplexExpand[Re[f[t]]];
v[t_] = ComplexExpand[Im[f[t]]]
i1 = ∫ f[t] dt
i2 = ∫ u[t] dt + i ∫ v[t] dt
Simplify[i1 - i2]
```

$$\frac{1}{4}(-i + t)^4$$

$$-\frac{3t^2}{2} + \frac{t^4}{4} + i(t - t^3)$$

```
1/4
```

Функция ComplexExpand[$expr$] выполняет разложение выражения `expr` в предположении, что все переменные вещественные. Как можно было ожидать, первообразные, полученные разными путями, отличаются только константой.

Аналогично можно вычислять определенные интегралы.

$$\int_0^2 f[t]\,dt$$

$-2-6i$

или

$$\int_0^2 u[t]\,dt + I\int_0^2 v[t]\,dt$$

$-2-6i$

Оба способа приводят к одинаковому результату.

Функция `Integrate` умеет вычислять интегралы по ломаным в комплексной плоскости. Для этого используется форма Integrate$[f[z],\{z,z_1,z_2,\ldots,z_k\}]$, где z_i представляют комплексные числа (точки комплексной плоскости) и интегрирование выполняется по отрезкам от точки z_i до точки z_{i+1}. Например,

$$\mathbf{Integrate}[\frac{1}{z},\{z,1+I,-1+I,-1-I,1-I,1+I\}]$$

$2i\pi$

Тот же результат можно получить, суммируя интегралы по отрезкам.

$$\mathbf{mi} = \{\mathbf{Integrate}[\frac{1}{z},\{z,1+I,-1+I\}],\mathbf{Integrate}[\frac{1}{z},\{z,-1+I,-1-I\}],$$
$$\mathbf{Integrate}[\frac{1}{z},\{z,-1-I,1-I\}],\mathbf{Integrate}[\frac{1}{z},\{z,1-I,1+I\}]\}$$

$$\left\{\frac{i\pi}{2},\frac{i\pi}{2},\frac{i\pi}{2},\frac{i\pi}{2}\right\}$$

$$\mathbf{Total}[\mathbf{mi}]$$

$2\,i\,\pi$

Функция `Total[list]` возвращает сумму элементов списка `list`.

Вот еще пример интегрирования функции по ломаной в комплексной плоскости.

$$\mathbf{Integrate}[\frac{1}{z+1/2},\{z,1,e^{\frac{i\pi}{3}},e^{\frac{2i\pi}{3}},-1,e^{-\frac{2i\pi}{3}},e^{-\frac{i\pi}{3}},1\}]//\mathbf{FullSimplify}$$

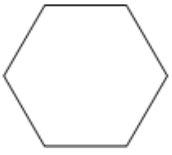
$2i\pi$

Можно нарисовать контур интегрирования, например, так

$$\mathbf{pp} = \{\mathbf{Re}[\#],\mathbf{Im}[\#]\}\&/@\mathbf{Exp}[\frac{2\pi i}{6}\mathbf{Range}[0,6]]$$

$$\{\{1,0\},\{\frac{1}{2},\frac{\sqrt{3}}{2}\},\{-\frac{1}{2},\frac{\sqrt{3}}{2}\},\{-1,0\},\{-\frac{1}{2},-\frac{\sqrt{3}}{2}\},\{\frac{1}{2},-\frac{\sqrt{3}}{2}\},\{1,0\}\}$$

$$\mathbf{Graphics@Line@(pp)}$$



Функцию `Integrate` можно использовать для вычисления интегралов по кривым в комплексной плоскости. Если функция $f(z)$ определена на контуре C комплексной плоскости, который имеет параметрическое уравнение $z = z(t), a \le t \le b$, то имеет место формула

$$\int_C f(z)\mathrm{d}z = \int_a^b f(z(t))z'(t)\,dt$$

Вычислим интеграл функции $f(z) = \dfrac{1}{z+1/2}$ по единичной окружности. Ее параметрическое уравнение имеет вид $z(t) = e^{it}$, $0 \le t \le 2\pi$. Тогда

```
f[z_] = 1/(z + 1/2);
z[t_] = Exp[I t];
F[t_] = f[z[t]] z'[t]
```

$$\frac{i e^{it}}{\frac{1}{2} + e^{it}}$$

```
Integrate[F[t], {t, 0, 2π}]
```

$2\,i\,\pi$

3.3.2 Численное интегрирование

Когда *Mathematica* не может взять определенный интеграл, она возвращает команду обратно точно также, как и при вычислении неопределенного интеграла

```
Integrate[Cos[10 Cos[x]], {x, 0, 1}]
```

$$\int_0^1 \mathrm{Cos}[10\,\mathrm{Cos}[x]]\,dx$$

Вы можете использовать функцию N для численного интегрирования

```
N[Integrate[Cos[10 Cos[x]], {x, 0, 1}]]
```

-0.301928

или

```
N[Integrate[Cos[10 Cos[x]], {x, 0, 1}], 20]
```

-0.30192779721155889037

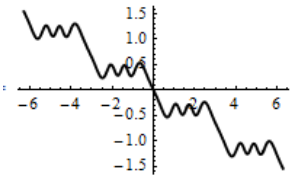
Если вы знаете, что хотите интегрировать численно, то вы можете использовать функцию `NIntegrate`. Это не разрешает *Mathematica* пробовать вычислять интеграл в символьном виде

```
NIntegrate[Cos[10 Cos[x]], {x, 0, 1}]
```

-0.301928

Вы можете подумать, что неопределенный интеграл не может быть получен численно, потому что ответом является функция, а не число. Но это не так. Построим, например, график функции $F(x)=\int_0^x \cos(10\cos t)\,dt$, являющейся первообразной функции $\cos(10\cos x)$.

```
F[x_] := NIntegrate[Cos[10 Cos[t]], {t, 0, x}];
Plot[F[x], {x, -2π, 2π}]
```



Важным моментом в приведенном коде является отложенное присваивание при создании функции F[x]. Функция `Plot` при построении графика сначала выбирает значение переменной *x* из заданного интервала и подставляет ее в аргумент функции F[x]. Поскольку присваивание отложенное, то вычисление интеграла происходит численно при этом значении *x* и получается точка графика. Затем переменной *x* функция `Plot` назначает другое значение и вычисление интеграла происходит при этом новом значении. Каждый раз интеграл вычисляется численно при новом значении верхнего предела *x*. Затем из полученного множества точек функция `Plot` строит график.

Для построения графика первообразной можно поступить по – другому. Аналогично тому, как в *Mathematica* приближаются числа, вы можете приблизить функцию F(*x*) другой функцией. Для этого нужно вычислить значение F(*x*) в некотором наборе точек и построить интерполяционную функцию. Эту последовательность действий выполняет функция `NDSolve`, которая в общем случае численно решает дифференциальное уравнение. В следующем примере мы построим график первообразной функции $\cos(10\cos x)$ в интервале от -10 до 10.

```
ss = NDSolve[{y'[x] == Cos[10 Cos[x]], y[0] == 0}, y[x], {x, -2π, 2π}]
{{y[x] → InterpolatingFunction[{{-6.28319,6.28319}}, " <> "][x]}}
Plot[y[x]/. ss, {x, -2π, 2π}]  (* предыдущий график *)
```

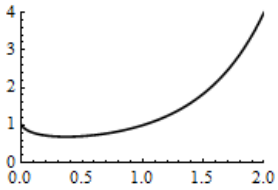
Здесь функция `NDSolve` построила приближенное решение дифференциального уравнения $y'(x)=\cos(10\cos x)$ с начальным условием $y(0)=0$ в виде интерполяционной функции `InterpolatingFunction`.

О функции `NDSolve` мы будем говорить подробно в следующей главе. Вы можете использовать последний пример в качестве шаблона. Замените Cos[10 Cos[x]] вашей функцией, а интервал от -10 до 10 на такой, где вы хотите получить результат.

Основное назначение функции `NIntegrate` состоит в приближенном вычислении определенных интегралов – однократных, повторных или (много)кратных.

Ее можно использовать для вычисления интегралов по конечному отрезку вещественной оси.

NIntegrate$[x^x, \{x, 0, 2\}]$
Plot[x^x, {x, 0, 2}, **PlotRange** → {{0, 2}, {0, 4}}]
2.83388



Можно численно находить интеграл на бесконечном участке вещественной оси

NIntegrate$\left[\frac{1}{1+x^2}, \{x, 0, \infty\}\right]$
1.5708

Численное интегрирование можно выполнять по конечному или бесконечному отрезку в комплексной плоскости.

NIntegrate$[\sqrt{x}, \{x, I, 3-I\}]$
$3.79214 - 2.21131i$

Обычно второй аргумент функции `NIntegrate` имеет вид короткого списка `{x, xmin, xmax}`. Но для подынтегральных выражений с особенностями его можно задавать в виде `{x, x0, x1, ..., xn}`. В этом случае выполняется проверка наличия особенности в каждой из внутренних точек x_i. Если особенностей нет, то результат эквивалентен интегралу от x_0 до x_n.

NIntegrate$[1/\mathbf{Sqrt}[x], \{x, -1, 1\}]$
Numerical integration converging too slowly...

NIntegrate$[1/\mathbf{Sqrt}[x], \{x, -1, 0, 1\}]$
2. - 2. i

Числа `x, x0, x1, ..., xn` могут быть комплексными и тогда интеграл будет представлять интеграл по ломаной в комплексной плоскости.

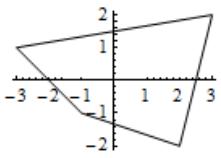
$f[z\_] = \frac{1 - e^z + z}{z^3(z-1)^2}$;

Chop[**NIntegrate**$[f[z], \{z, 2-2I, 3+2I, -3+I, -1-I, 2-2I\}]$]
$0. - 0.398582\, i$

Функция `Chop[expr]` заменяет нулем в выражении `expr` все близкие к нулю числа, поскольку обычно они являются погрешностью вычислений.

Контур, по которому выполнено интегрирование, показан на следующем рисунке.

pp = {**Re**[#], **Im**[#]}&/@{$2-2I, 3+2I, -3+I, -1-I, 2-2I$}
Show[**Graphics@Line@pp**, **Axes** → **True**]
$\{2,-2\},\{3,2\},\{-3,1\},\{-1,-1\},\{2,-2\}\}$



Можно численно интегрировать по контуру на комплексной плоскости. В следующем примере вычисляется интеграл $\oint_C \frac{1}{z+1} dz$, где C – окружность радиуса 2 с центром в начале координат.

```
Chop[NIntegrate[ⅈ 2 Exp[ⅈ t]/(2 Exp[ⅈ t] + 1), {t, 0, 2π}]]
```

```
0.+6.2831853 i
```

Можно интегрировать списки выражений

```
NIntegrate[{x, x^2, x^3}, {x, 0, 1}]
```

```
{0.5,0.33333333,0.25}
```

Функция `NIntegrate` умеет вычислять повторные интегралы.

```
NIntegrate[Exp[-x^2 + y], {x, 0, ∞}, {y, 0, 1}]
```

```
1.5227876
```

Внутренний список $\{x, 0, \infty\}$ определяет внешний интеграл. Последним указывается интервал изменения той переменной, интегрирование по которой выполняется в первую очередь. При этом пределы интегрирования могут зависеть от переменной внешнего интеграла. Например, площадь единичного круга можно вычислить с помощью следующего интеграла

```
NIntegrate[1, {x, -1, 1}, {y, -Sqrt[1 - x^2], Sqrt[1 - x^2]}]
```

```
3.1415927
```

В следующем примере мы вычисляем трехкратный интеграл с особенностью в начале координат

```
NIntegrate[1/Sqrt[x^2 + y^2 + z^2], {x, 0, 1}, {y, 0, 1}, {z, 0, 1}]
```

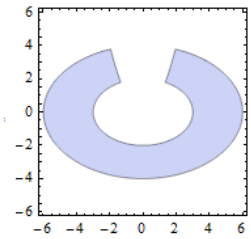
```
1.19004
```

Можно выполнять численное интегрирование по области. В следующем примере мы вычисляем площадь области эллиптического кольца с параболическим вырезом.

```
NIntegrate[Boole[1 ≤ x^2/9 + y^2/4 ≤ 4 && y < x^2], {x, -6, 6}, {y, -6, 6}]
```

```
RegionPlot[1 ≤ x^2/9 + y^2/4 ≤ 4 && y < x^2, {x, -6, 6}, {y, -6, 6}]
```

```
49.7429
```



У функции `NIntegrate` имеются опции, которые управляют точностью вычислений.

`AccuracyGoal` определяет абсолютную погрешность вычислений, используя количество значащих цифр. Значение по умолчанию

AccuracyGoal → ∞ говорит, что эта опция не должна использоваться как критерий прекращения вычислений (т.е. будут использоваться другие опции).

Когда значение интеграла равно нулю функция NIntegrate возвращает сообщение о медленной сходимости алгоритма. В такой ситуации требуется установка опции AccuracyGoal в значение, отличное от установленного по умолчанию (от Infinity) и критерием остановки вычислений будет абсолютная погрешность.

NIntegrate[Sin[x], {x, 0, 2π}]

```
Numerical integration converging too slowly; suspect one of the
following: singularity, value of the integration is 0, …
```

Но

NIntegrate[Sin[x], {x, 0, 2π}, AccuracyGoal → 6]

```
-2.886686×10^-16
```

PrecisionGoal определяет «относительную погрешность» вычислений и задается количеством значащих цифр; PrecisionGoal → ∞ указывает, что эта опция не будет использоваться как критерий прекращения вычислений (будет использоваться опция AccuracyGoal). Для опции PrecisionGoal мы используем не совсем корректный термин «относительная погрешность». PrecisionGoal->n определяет для значения *x* погрешность вычислений равную $|x|10^{-n}$. Когда включены опции AccuracyGoal->m и PrecisionGoal->n, то *Mathematica* пытается выполнять вычисления величины *x* с погрешностью не более $10^{-m}+|x|10^{-n}$.

Опция WorkingPrecision определяет количество значащих цифр, используемых во внутренних вычислениях; значение этой опции, как правило, должно быть больше значения опции AccuracyGoal; установка WorkingPrecision->MachinePrecision приводит к вычислениям с процессорной точностью.

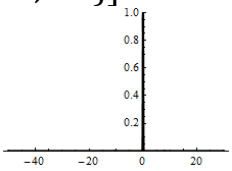
NIntegrate[$\frac{1}{1+\sqrt{x}}$, {x, 0, 1}, PrecisionGoal → 20, WorkingPrecision → 40]

```
0.61370563888010938116553575708364686389 90
```

Функция NIntegrate выполняет разбиение участка интегрирования до тех пор, пока ошибка вычислений выбранного ею метода не станет меньше погрешности, определяемой опциями AccuracyGoal и PrecisionGoal. Функция NIntegrate использует адаптивные алгоритмы с автоматическим выбором шага интегрирования. Эти алгоритмы сами, если это требуется, рекурсивно дробят отрезки разбиения области интегрирования в местах быстрого изменения функции. Опция MinRecursion, определяет количество шагов дробления отрезков разбиения, с которого начинаются вычисления. Опция MaxRecursion определяет максимальное количество шагов дробления.

Рассмотрим пример. Функции NIntegrate может пропустить участки быстрого возрастания подынтегрального выражения. Вот пример такой функции.

$$\mathbf{Plot}[\mathbf{Exp}[-\mathbf{100}\boldsymbol{x}^{\mathbf{2}}],\{\boldsymbol{x},-\mathbf{50},\mathbf{30}\},\mathbf{PlotRange}\rightarrow\{\mathbf{0},\mathbf{1}\},\mathbf{PlotPoints}\rightarrow\mathbf{1000}]$$
$$\mathbf{NIntegrate}[\mathbf{Exp}[-\mathbf{100}\boldsymbol{x}^{\mathbf{2}}],\{\boldsymbol{x},-\mathbf{50},\mathbf{30}\}]$$




```
NIntegrate failed to converge to prescribed accuracy after 9
recursive bisections ...
```

Увеличив значение опция `MinRecursion`, можно вынудить функцию `NIntegrate` в самом начале делать более мелкое дробление участков разбиения.

$$\mathbf{NIntegrate}[\mathbf{Exp}[-\mathbf{100}\boldsymbol{x}^{\mathbf{2}}],\{\boldsymbol{x},-\mathbf{50},\mathbf{30}\},\mathbf{MinRecursion}\rightarrow\mathbf{4}]$$

```
0.177245
```

По умолчанию установлено `MinRecursion->0`. Опция `MaxRecursion→n` определяет, что рекурсивно можно дробить отрезки разбиения n раз. Рекурсивное/многократное деление выполняется только на тех участках, где значения функции быстро изменяются и для достижения требуемой точности длина отрезков разбиения должна быть уменьшена.

Опция `MaxPoints` ограничивает количество точек разбиения интервала интегрирования.

Опция `Exclusions` позволяет исключить из рассмотрения некоторые особые точки (кривые или поверхности для кратных интегралов) из области интегрирования.

$$\mathbf{NIntegrate}[\mathbf{Log}[\boldsymbol{x}-\mathbf{2}],\{\boldsymbol{x},\mathbf{0},\mathbf{4}\}]$$

```
Numerical integration converging too slowly; suspect one of the
following: singularity,...
```

$$\mathbf{NIntegrate}[\mathbf{Log}[\boldsymbol{x}-\mathbf{2}],\{\boldsymbol{x},\mathbf{0},\mathbf{4}\},\mathbf{Exclusions}\rightarrow\{\mathbf{2}\}]$$

```
-1.2274113+6.2831853 i
```

В функции `NIntegrate` реализовано много различных алгоритмов численного интегрирования. Обычно `NIntegrate` сама выбирает метод. Однако иногда вы будете задавать метод с помощью опции `Method`. Мы не ставим перед собой задачу изложения всех методов, используемых этой функцией. С ними вы можете познакомиться по справочной системе.

В частности, если вы хотите численно найти главное значение интеграла, то следует использовать опцию `Method→"PrincipalValue"`.

$$\mathbf{NIntegrate}[\frac{\mathbf{1}}{\boldsymbol{x}},\{\boldsymbol{x},-\mathbf{1},\mathbf{2}\},\mathbf{Method}\rightarrow\text{"}\mathbf{PrincipalValue}\text{"},\ \mathbf{Exclusions}\rightarrow\boldsymbol{x}==\mathbf{0},\mathbf{AccuracyGoal}\rightarrow\mathbf{8}]$$

```
0.69314718
```

Попробуйте удалить хотя бы одну из опций предыдущего примера и вы не поучите результат или получите предупреждение системы о медленной сходимости.

Опцией `EvaluationMonitor` задается выражение, которое вычисляется каждый раз, когда вычисляется подынтегрального выражение. В

следующем примере определяется количество точек, использованных при численном интегрировании

```
Block[{k = 0}, {NIntegrate[x^3, {x, 0, 2}, EvaluationMonitor :> k++], k}]
{4.,11}
```

Функция `Block[{x=x0,y=y0,...},expr]` выполняет блок кода с локальными переменными `x,y,...`, для которых заданы начальные значения x_0, y_0 и т.д.

Значение интеграла равно 4 и использовано всего 11 точек. Можно даже узнать эти точки.

```
Block[{X = {}}, {NIntegrate[x^3, {x, 0, 2},
                  EvaluationMonitor :> (X = Join[X, {x}])], X}]
{4.,
{0.01591464,0.093820154,0.24583327,0.46153069,0.72036959,
1.,1.2796304,1.5384693,1.7541667,1.9061798,1.9840854}}
```

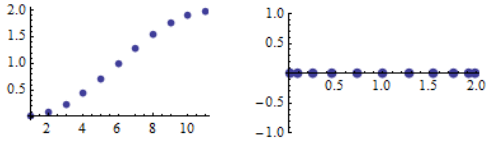
Те же действия можно выполнить командой

```
r = Reap[NIntegrate[x^3, {x, 0, 2}, EvaluationMonitor:> Sow[x]]]
{4.,
{{0.01591464,0.093820154,0.24583327,0.46153069,0.72036959
,1.,1.2796304,1.5384693,1.7541667,1.9061798,1.9840854}}}
```

Функция `Sow[`*выражение*`]` «сеет» значения *выражения*, которые «собираются» ближайшей охватывающей функцией `Reap[expr]`, которая также возвращает значение выражения `expr`.

Можно построить график значений этих точек (следующий рисунок слева)

```
ListPlot[r[[2, 1]], PlotRange-> All, PlotStyle-> PointSize[0.02]]
```



или показать, как эти точки расположены на отрезке интегрирования (предыдущий рисунок справа)

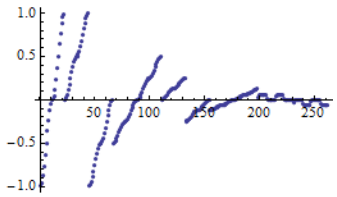
```
pp = Transpose[{r[[2, 1]], ConstantArray[0, Length[r[[2, 1]]]]}]
{{0.01591464,0},{0.093820154,0},{0.24583327,0},
 {0.46153069,0},{0.72036959,0},...,{1.9840854,0}}
ListPlot[pp, PlotStyle-> PointSize[0.03]]
```

Для несобственного интеграла точек разбиения много больше

```
Block[{k = 0}, {NIntegrate[1/Sqrt[x], {x, 0, 1}, EvaluationMonitor :> k++], k}]
{2.,131}
```

Следующий график поясняет как функция `NIntegrate` размещает точки разбиения

```
rp = Reap[NIntegrate[1/Sqrt[x], {x, -1, 0, 1},
                     EvaluationMonitor:> Sow[x]]];
ListPlot[rp[[2, 1]], PlotRange-> All]
```



По горизонтали отложены номера точек, а по вертикали – координаты точек на оси X. По графику можно проанализировать стратегию выбранного функцией `NIntegrate` метода. Первые ≈ 20 точек размещаются более менее равномерно на отрезке [-1, 1]. Следующие ≈ 50 точек размещаются на отрезках интегрирования, образуемых первыми точками. На следующем шаге функция `NIntegrate` разбивает отрезки из интервала (-0.5, 0.5). Потом разбиваются отрезки из интервала (-0.25, 0.25). Далее функция `NIntegrate` выполняет еще несколько шагов, дробя отрезки интегрирования, расположенные в окрестности нуля.

3.3.3 Вычисление длин, площадей и объемов

Длина кривой.

Длина дуги плоской кривой, заданной явным уравнением $y = y(x)$ в декартовых координатах, вычисляется по формуле

$$L = \int_{x_0}^{x_1} \sqrt{1 + (y'(x))^2}\, d\,x, \qquad (1)$$

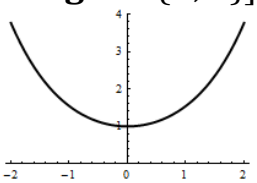
где x_0 и x_1 определяют начальную и конечную точки дуги.

Пример. Длина цепной линии $y = a\,ch\frac{x}{a}$.

```
Clear[a, x, y, t];
y[x_] = a Cosh[x/a];
L = Integrate[Sqrt[1 + D[y[t], t]^2], {t, 0, x}, Assumptions :> {x ∈ Reals, a > 0}]
```

$$a \operatorname{Sinh}\left[\frac{x}{a}\right]$$

```
a = 1; Plot[y[x], {x, -2, 2}, PlotRange → {0, 4}]
```



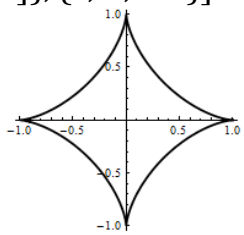
Длина дуги плоской кривой, заданной в параметрическом виде $x = x(t),\ y = y(t)$, вычисляется по формуле

$$L = \int_{t_0}^{t_1} \sqrt{(x'(t))^2 + (y'(t))^2}\, d\,t \qquad (2)$$

где t_0 и t_1 – значения параметра t в начальной и конечной точках дуги.

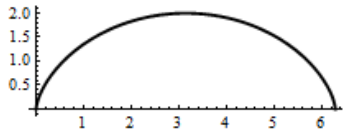
Пример. Длина астроиды $x = a\cos^3 t,\ y = a\sin^3 t,\quad 0 \le t \le 2\pi$.

```
Clear[a, x, y]
x[t_] = a Cos[t]^3;
y[t_] = a Sin[t]^3;
L = Integrate[Sqrt[D[x[t], t]^2 + D[y[t], t]^2], {t, 0, 2 π}, Assumptions :→ a > 0]
6 a
a = 1; ParametricPlot[{x[t], y[t]}, {t, 0, 2 π}]
```



Пример. Длина циклоиды $x = a(t - \sin t),\ y = a(1 - \cos t),\quad 0 \le t \le 2\pi$.

```
Clear[a]
x[t_] = a(t − Sin[t]);
y[t_] = a(1 − Cos[t]);
L = Integrate[Sqrt[D[x[t], t]^2 + D[y[t], t]^2], {t, 0, 2 π}, Assumptions :→ a > 0]
8 a
a = 1; ParametricPlot[{x[t], y[t]}, {t, 0, 2 π}]
```



Пример. Длина дуги эллипса.

```
Clear[a, b, x, y, t]
x[t_] = a Sin[t];
y[t_] = b Cos[t];
```

В следующей строке кода мы выполняем преобразование подынтегрального выражения $\sqrt{(x'(t))^2 + (y'(t))^2}$ к виду, с которым может «справиться» функция `Integrate` (вычитаем и добавляем a^2, делим на a^2 под корнем, умножаем корень на a, множитель $1 - b^2/a^2$ обозначаем p^2).

```
dl = a Simplify[Sqrt[(Simplify[D[x[t], t]^2 + D[y[t], t]^2 − a^2] + a^2)/a^2]] /. {(−1 + b^2/a^2) → −p^2}
a Sqrt[1 − p^2 Sin[t]^2]
```

Величина $p = \sqrt{1 - b^2/a^2}$ называется эксцентриситетом эллипса, если b меньшая его полуось. Вычисляя длину дуги эллипса от верхнего конца малой оси до любой его точки в первом квадранте, получим

```
Integrate[dl, {t, 0, θ}, Assumptions :→ {0 < p < 1, 0 < θ < π}] /. p^2 → 1 − b^2/a^2
a EllipticE[θ, 1 − b^2/a^2]
```

Здесь мы вернулись к прежним обозначениям. Т.о. длина дуги эллипса выражается эллиптическим интегралом 2-го рода; этот факт послужил поводом для самого названия «эллиптический».

Длина всего обвода эллипса равна

$$\boldsymbol{L = \text{Integrate}[\text{dl}, \{t, 0, 2\pi\}, \text{Assumptions} :\to \{0 < p < 1\}]/.\, p^2 \to 1 - \frac{b^2}{a^2}}$$

$4\,a\,\text{EllipticE}\left[1 - \frac{b^2}{a^2}\right]$ (* полный эллиптический интеграл *)

Численное значение длины эллипса получается использованием функции N.

$$\boldsymbol{b = 1; a = 2; N[L]}$$

```
9.6884482
```

Длина дуги кривой, заданной в полярных координатах уравнением $r = r(\varphi)$, вычисляется по формуле

$$L = \int_{\varphi_0}^{\varphi_1} \sqrt{r^2(\varphi) + (r'(\varphi))^2}\, d\varphi \qquad (3)$$

где φ_0 и φ_1 – значения углового параметра φ в начальной и конечной точках дуги.

Пример. Длина логарифмической спирали $r = a e^{m\varphi}$.

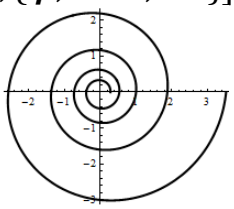
$$\boldsymbol{\text{Clear}[a, m]}$$

$$\boldsymbol{r[\varphi\_] = a\ \text{Exp}[m\ \varphi];}$$

$$\boldsymbol{L = \text{Integrate}[\sqrt{r[\varphi]^2 + D[r[\varphi], \varphi]^2}, \{\varphi, 0, \theta\}, \text{Assumptions} :\to \{a > 0, m > 0\}]}$$

$$\frac{a(-1 + e^{m\theta})\sqrt{1 + m^2}}{m}$$

$$\boldsymbol{a = 1; m = 0.1;\ \text{PolarPlot}[r[\varphi], \{\varphi, -4\pi, 4\pi\}]}$$



Длина дуги пространственной кривой, заданной параметрически $x = x(t), y = y(t), z = z(t)$, вычисляется по формуле

$$L = \int_{t_0}^{t_1} \sqrt{(x'(t))^2 + (y'(t))^2 + (z'(t))^2}\, dt \qquad (4)$$

где t_0 и t_1 – значения параметра t в начальной и конечной точках дуги.

Пример. Длина винтовой линии $x = a\cos t$, $y = a\sin t$, $z = ct$ от точки $t = 0$ до точки $t = T$ будет равна

$$\boldsymbol{\text{Clear}[a, c, x, y, z]}$$

$$\boldsymbol{\{x[\text{t\_}], y[\text{t\_}], z[\text{t\_}]\} = \{a\ \text{Cos}[t], a\ \text{Sin}[t], ct\};}$$

$$\boldsymbol{L = \text{Integrate}[\sqrt{D[x[t], t]^2 + D[y[t], t]^2 + D[z[t], t]^2}, \{t, 0, T\}]}$$

$$\sqrt{a^2 + c^2}T$$

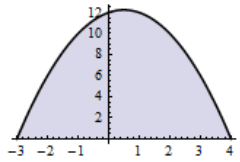
Площадь плоской фигуры.

Геометрический смысл определенного интеграла $\int_a^b f(x)dx$ состоит в том, что он представляет *площадь под кривой* $y=f(x)$ при $a\le x\le b$.

Пример. Найдем площадь фигуры, ограниченной сверху параболой $y=9-x^2$, а снизу осью абсцисс. Для этого решим уравнение $12+x-x^2=0$ и найдем точки, в которых парабола пересекается с осью абсцисс.

```
y[x_] = -x^2 + x + 12;
{x1, x2} = x /. Solve[y[x] == 0, x]
Plot[y[x], {x, x1, x2}, Filling → Axis]
```

{-3,4}

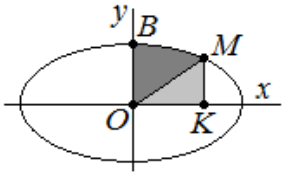


Теперь вычисляем площадь

```
S = Integrate[y[x], {x, x1, x2}]
```

$$\frac{343}{6}$$

Пример. Даны эллипс $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$ и точка $M(x,y)$ на нем. Определить площадь криволинейной трапеции BOKM и сектора OMB.



Из уравнения эллипса имеем

```
y[x_] = b/a Sqrt[a^2 - x^2];
SBOKM = Apart[Integrate[y[t], {t, 0, x},
                Assumptions :→ {a > 0, b > 0, 0 < x < a}]]
```

$$\frac{b\,x\,\sqrt{a^2-x^2}}{2\,a}+\frac{1}{2}a\,b\,\text{ArcSin}\left[\frac{x}{a}\right]$$

Но

```
SBOKM/.{b/a Sqrt[a^2 - x^2] → y}
```

$$\frac{x\,y}{2}+\frac{1}{2}a\,b\,\text{ArcSin}[\frac{x}{a}]$$

Так как первое слагаемое представляет площадь ΔOKM, то, отнимая ее, для площади сектора получим выражение $S_{OMB}=\frac{ab}{2}\arcsin\frac{x}{a}$. При $x=a$ для площади четверти эллипса находим значение $\frac{\pi ab}{4}$.

Площадь области D на плоскости xOy, заданной неравенствами или неявно, *можно находить с помощью двойного интеграла*

$$S = \iint\limits_D d\,x\,d\,y\,.$$

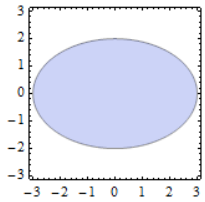
Пример 2. Площадь эллипса $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$.

$\mathbf{rgn} = \frac{\boldsymbol{x^2}}{\mathbf{9}} + \frac{\boldsymbol{y^2}}{\mathbf{4}} \leq \mathbf{1};$

$\mathbf{Integrate[Boole[rgn], \{x, -3, 3\}, \{y, -2, 2\}]}$

6π

$\mathbf{RegionPlot[rgn, \{x, -3, 3\}, \{y, -3, 3\}]}$



Пример. Площадь повернутого эллипса $a\,x^2+2\,b\,x\,y+c\,y^2=1\left(a\,c-b^2>0, c>0\right)$.

$\mathbf{Clear[a, b, c];}$

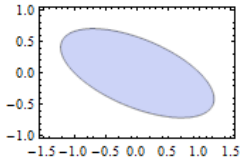
$\mathbf{Integrate[Boole[ax^2 + 2bxy + cy^2 \leq 1], \{x, -\infty, \infty\}, \{y, -\infty, \infty\},}$
$\mathbf{Assumptions :\to \{c > 0, ac - b^2 > 0\}]}$

$$\frac{\pi}{\sqrt{-b^2+ac}}$$

Изобразим эллипс при некоторых значениях параметров a, b, c.

$\mathbf{a = 1; b = 1; c = 3;}$

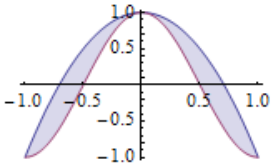
$\mathbf{RegionPlot[ax^2 + 2bxy + cy^2 \leq 1, \{x, -1.5, 1.5\}, \{y, -1., 1.\},}$
$\mathbf{AspectRatio \to Automatic]}$



Площадь между двумя кривыми в *Mathematica* можно вычислить несколькими способами: как однократный интеграл от их разности; как повторный интеграл по области, ограниченной этими кривыми; как двойной интеграл по области.

Пример 1. Вычислить площадь между кривыми $y_1(x)=1-2\,x^2$ и $y_2(x)=\cos\pi\,x$.

$\mathbf{Plot[\{1 - 2x^2, Cos[\pi x]\}, \{x, -1, 1\}, Filling \to 1 \to \{2\}]}$



Вычислим площадь между кривыми как однократный интеграл от их разности

$\mathbf{Integrate[1 - 2x^2 - Cos[\pi x], \{x, -1, 1\}]}$

$\frac{2}{3}$

Вычислим площадь между кривыми как повторный интеграл по области, ограниченной этими кривыми

```
Integrate[1, {x, -1, 1}, {y, Cos[πx], 1 - 2x^2}]
```

$$\frac{2}{3}$$

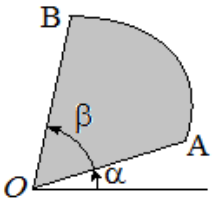
Вычислим площадь между кривыми как двойной интеграл

```
Integrate[Boole[Cos[πx] < y < 1 - 2x^2], {x, -1, 1}, {y, -1, 1}]
```

```
2/3
```

Площадь области в полярных координатах.

Дан сектор AOB, ограниченный кривой AB и двумя радиусами – векторами OA и OB (каждый из которых может свестись к точке). Кривая задается полярным уравнением $r = r(\varphi)$ $(\alpha \le \varphi \le \beta)$.



Площадь криволинейного сектора AOB вычисляется по формуле

$$S = \iint\limits_D d\,s = \iint\limits_D r\,d\,r\,d\,\varphi = \int\limits_\alpha^\beta d\,\varphi \int\limits_0^{r(\varphi)} r\,d\,r\,, \qquad (5)$$

Если вычислить внутренний интеграл, то получим

$$S = \frac{1}{2}\int\limits_\alpha^\beta r^2(\varphi) d\,\varphi\,. \qquad (6)$$

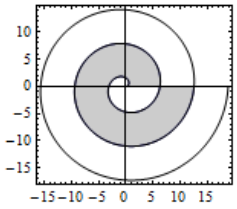
Пример. Вычислить площадь фигуры, заключенной между последовательными витками (первым и вторым) архимедовой спирали $r = a\varphi$. Очевидно, что она вычисляется как разность площадей второго и первого витков.

```
a =.;
r[φ_] = aφ;
s1 = Integrate[r[φ]^2, {φ, 0, 2π}]/2;     (* площадь 1-го витка *)
s2 = Integrate[r[φ]^2, {φ, 2π, 4π}]/2;   (* площадь 2-го витка *)
s2 - s1
```

$$8a^2\pi^3$$

Поскольку у функции `PolarPlot` нет опции `Filling`, то для представления области перейдем к ее параметрическому представлению

```
a = 1; r[φ_] = a φ;
rv1[t_] = {r[t] Cos[t], r[t] Sin[t]};  (* радиус-вектор первого витка спирали *)
rv2[t_] = rv1[t + 2π];                 (* радиус-вектор второго витка спирали *)
pa = ParametricPlot[{v rv1[t] + (1 - v)rv2[t]}, {t, 0, 2π}, {v, 0, 1},
                    Mesh → None];
pc = PolarPlot[r[φ], {φ, 0, 6π}];
Show[pa, pc]
```

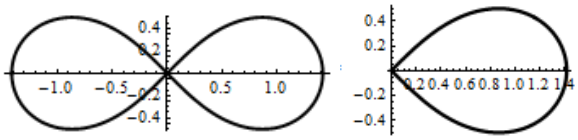


Заметим, что площадь **s1**, посчитанная выше, соответствует незакрашенной внутренней области рисунка.

□

Пример. Найти площадь лемнискаты $r^2 = 2a^2 Cos\,2\varphi$.

```
a = 1; r[φ_] = a√2 √Cos[2φ];
PolarPlot[r[φ], {φ, -π, π}]                    (* рисунок слева *)
PolarPlot[r[φ], {φ, -π/4, π/4},
        PlotStyle → {Black, Thickness[0.02]}]   (* рисунок справа *)
```



Правый рисунок показывает, что достаточно удвоить площадь правого овала, которому отвечает диапазон изменения угла $-\frac{\pi}{4} \le \varphi \le \frac{\pi}{4}$. Имеем

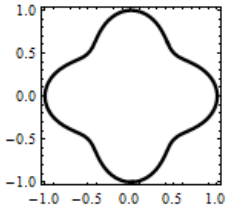
```
a =.; r[φ_] = a√2Cos[2φ];
s1 = Integrate[r[φ]^2, {φ, -π/4, π/4}]/2
```

a^2

Т.о. площадь лемнискаты (двух петель) равна $2a^2$.

Пример. Вычислим площадь фигуры, ограниченной кривой $\left(x^2+y^2\right)^3 = a^2\left(x^4+y^4\right)$. Легко видеть, что кривая симметрична относительно осей и точка $(0,0)$ является изолированной.

```
a = 1;
ContourPlot[(x^2 + y^2)^3 == a^2 (x^4 + y^4), {x, -1, 1}, {y, -1, 1}]
```



Вычисление площади как интеграла по области не дает точного результата.

```
Integrate[Boole[(x^2 + y^2)^3 <= a^2 (x^4 + y^4)], {x, -∞, ∞}, {y, -∞, ∞}]
```

$$\int_{-1}^{1} (-\text{Root}[-x^4 + x^6 + 3x^4\#1^2 + (-1+3x^2)\#1^4 + \#1^6\&,1] + \text{Root}[-x^4 + x^6 + 3x^4\#1^2 + (-1+3x^2)\#1^4 + \#1^6\&,2])\,dx$$

Хотя численное значение легко находится

```
N[%]
2.3561945
```

Переход к полярным координатам позволяет получить точный результат. Полярное уравнение кривой имеет вид $r^2 = a^2\left(\cos^4\varphi + \sin^4\varphi\right)$. Учитывая симметрию и используя формулу (5), получаем

$\boldsymbol{a} =.; \boldsymbol{r} =.;$

$\boldsymbol{S} = \mathbf{8}\ \mathbf{Integrate}[\boldsymbol{r}, \{\boldsymbol{\varphi}, \mathbf{0}, \boldsymbol{\pi}/\mathbf{4}\}, \{\boldsymbol{r}, \mathbf{0}, \boldsymbol{a}\sqrt{\mathbf{Cos}[\boldsymbol{\varphi}]^{\mathbf{4}} + \mathbf{Sin}[\boldsymbol{\varphi}]^{\mathbf{4}}}\}]$

$\frac{3a^2\pi}{4}$

Площадь плоской области, контур которой задан параметрически.

Площадь области D, ограниченной кусочно – гладкой замкнутой кривой L, можно вычислять с помощью следующих криволинейных интегралов:

$$S = \oint_L x\,d\,y, \tag{7}$$

$$S = -\oint_L y\,d\,x. \tag{8}$$

Чаще для вычисления площади применяют более симметричную формулу

$$S = \frac{1}{2}\oint_L x\,d\,y - y\,d\,x \tag{9}$$

Во всех формулах выбирается положительное направление обхода контура *L*.

Для проверки формул (7) – (9) вспомним формулу Грина. Если функции $P = P(x, y), Q = Q(x, y)$ определены в области *D* и имеют непрерывные частные производные $\frac{\partial P}{\partial y}, \frac{\partial Q}{\partial x}$, то имеет место равенство

$$\iint_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) d\,x\,d\,y = \oint_L P\,d\,x + Q\,d\,y$$

Если $Q = x, P = 0$, то получим $\iint_D dx\,dy = \oint_L x\,dy$. Но слева стоит двойной интеграл по области, который представляет ее площадь S. Аналогично, полагая $Q = 0, P = -y$ или $Q = x, P = -y$, приходим к (8) и (9).

Пусть теперь параметрическое уравнение кривой *L*, являющейся границей области *D*, имеет вид $x = x(t), y = y(t), \ t_{\min} \le t \le t_{\max}$, тогда формула (7) принимает вид

$$S = \int_{t_{\min}}^{t_{\max}} x(t)\,y'(t)\,d\,t \tag{10}$$

Аналогично записываются другие формулы для площади области, если контур *L* задан в параметрическом виде.

Пример. Вычислим площадь эллипса.

$\{\boldsymbol{x}, \boldsymbol{y}\} = \{\boldsymbol{a}\ \mathbf{Cos}[\boldsymbol{t}], \boldsymbol{b}\ \mathbf{Sin}[\boldsymbol{t}]\};$

$\mathbf{Integrate}[\boldsymbol{xD}[\boldsymbol{y}, \boldsymbol{t}], \{\boldsymbol{t}, \mathbf{0}, \mathbf{2}\boldsymbol{\pi}\}]$

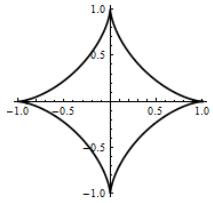
a b π

$\boldsymbol{a} = \mathbf{3}; \boldsymbol{b} = \mathbf{2};$

$\mathbf{ParametricPlot}[\{\boldsymbol{x}, \boldsymbol{y}\}, \{\boldsymbol{t}, \mathbf{0}, \mathbf{2}\boldsymbol{\pi}\}]$

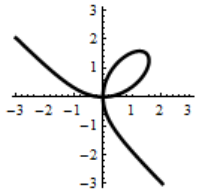
Пример. Вычислим площадь астроиды по формуле (9)

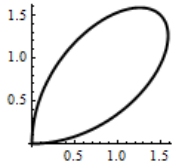
```
x[t_] = a Cos[t]^3;
y[t_] = a Sin[t]^3;
x[t]Dt[y[t], t, Constants → a]
y[t]Dt[x[t], t, Constants → a]
  3a^2Cos[t]^4Sin[t]^2
-3a^2Cos[t]^2Sin[t]^4
S = 1/2 Integrate[x[t]Dt[y[t], t, Constants → a] -
                  y[t]Dt[x[t], t, Constants → a], {t, 0, 2π}]
3a^2π/8
a = 1;  ParametricPlot[{x[t], y[t]}, {t, 0, 2π}]
```



Пример. Найти площадь петли декартова листа $x^3 + y^3 = 3axy$.

```
a = 1;
ContourPlot[x^3 + y^3 == 3 a x y, {x, -3, 3}, {y, -3, 3},
            Axes → True, Frame → False]     (* след. рисунок слева *)
```





Параметрические уравнения контура петли можно записать в виде $x = \frac{2at}{1+t^3}$, $y = \frac{2at^2}{1+t^3}$. Следующее построение показывает, что петля описывается при изменении параметра *t* от 0 до ∞.

```
x[t_] = 3at/(1 + t^3);
y[t_] = 3at^2/(1 + t^3);
ParametricPlot[{x[t], y[t]}, {t, 0, 100}]  (* предыдущий рисунок справа *)
```

Для вычисления площади используем формулу (9). Имеем

```
a =.;   x[t_] = 3 a t/(1+t^3);  y[t_] = 3 a t^2/(1+t^3);
Simplify[Integrate[x[t]D[y[t], t] - y[t]D[x[t], t], {t, 0, ∞}]]/2
3a^2/2
```

Отметим, что здесь мы использовали несобственный интеграл с бесконечными пределами.

Объем тела.

Объем тела по площади параллельных сечений.

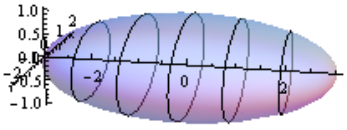
Рассмотрим некоторое тело (V), содержащееся между плоскостями $x=a$ и $x=b$, и станем рассекать его плоскостями, перпендикулярными к оси x. Допустим, что все сечения квадрируемы, и пусть площадь сечения $P(x)$, отвечающего абсциссе x, будет непрерывной функцией от x (для $a \le x \le b$). Пусть также любые два различных сечения, будучи спроектированы на плоскость, перпендикулярную к оси x, оказываются всегда содержащимися одно в другом. Тогда тело (V) имеет объем, который выражается формулой

$$V = \int_a^b P(x)\,dx \qquad (11)$$

Пример. Найти объем трехосного эллипсоида $\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1$.

Плоскость, перпендикулярная к оси x и проходящая через точку $M(x,0,0)$, пересечет эллипсоид по эллипсу (см. следующий рисунок).

```
a = 3; b = 2; c = 1;
x[u_, v_] = a Cos[u] Cos[v];
y[u_, v_] = b Cos[u] Sin[v];
z[u_, v_] = c Sin[u];
ParametricPlot3D[{x[u, v], y[u, v], z[u, v]}, {u, -π/2, π/2}, {v, 0, 2π},
    Boxed → False, AxesOrigin → {-a, 0, 0}, PlotStyle → Opacity[0.5],
    MeshFunctions → {#1&}, Mesh → 5]
```



Уравнение проекции этого эллипса на плоскость yOz будет таково

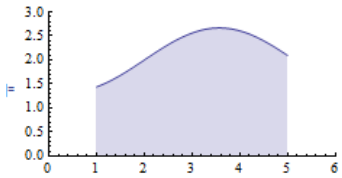
$$\frac{y^2}{b^2\left(1-x^2/a^2\right)}+\frac{y^2}{c^2\left(1-x^2/a^2\right)}=1 \quad \left(x=const\right).$$

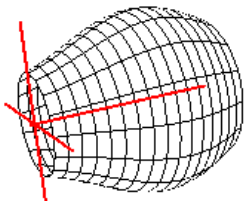
Отсюда ясно, что полуоси его будут, соответственно $b\sqrt{1-x^2/a^2}$ и $c\sqrt{1-x^2/a^2}$, а площадь выразится так $P(x)=\pi b c\left(1-x^2/a^2\right)$. Таким образом, имеем

```
Clear[a, b, c];
P[x_] = π b c/a^2 (a^2 - x^2);
V = ∫_-a^a P[x] dx
4/3 a b c π
```

Объем тела вращения.

Частный случай формулы (11), когда заведомо выполняется указанное выше предположение о взаимном расположении сечений, представляют тела вращения. Пусть на плоскости xOy задана кривая уравнением $y = f(x)$ $(a \le x \le b)$, где $f(x)$ непрерывна и неотрицательна. Станем вращать ограниченную ею криволинейную трапецию вокруг оси x (см. следующий рисунок).





Сечения полученного тела (V) удовлетворяют указанному предположению, поскольку проектируются на плоскость yOz в виде концентрических кругов. Здесь $P(x) = \pi y^2 = \pi (f(x))^2$, так что

$$V = \pi \int_a^b y^2 d\,x = \pi \int_a^b (f(x))^2 d\,x \qquad (12)$$

Выражение $V = \pi \int_a^b y^2 d\,x$ можно использовать и в случае, когда кривая задана параметрически.

Если криволинейная трапеция ограничена снизу и сверху кривыми $y_1 = f_1(x)$ и $y_2 = f_2(x)$, то, очевидно

$$V = \pi \int_a^b \left(y_2^{\,2} - y_1^{\,2}\right) d\,x = \pi \int_a^b \left((f_2(x))^2 - (f_1(x))^2\right) d\,x, \qquad (13)$$

хотя предположение о взаимном расположении сечений в этом случае может нарушаться.

Пример. Пусть эллипс $\dfrac{x^2}{a^2} + \dfrac{y^2}{b^2} = 1$ вращается вокруг оси x. Тогда, так как $y^2 = \dfrac{b^2}{a^2}\left(a^2 - x^2\right)$, для объема эллипсоида вращения имеем

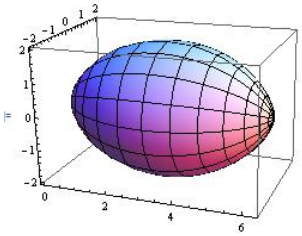
$$\boldsymbol{V = \pi \int_{-a}^{a} \frac{b^2}{a^2} (a^2 - x^2)\, dx}$$

$$\frac{4}{3} a\, b^2\, \pi$$

□

Пример. Определим объем тела, полученного вращением ветви циклоиды $x = a(t - \sin t)$, $y = a(1 - \cos t)$ $(0 \le t \le 2\pi)$ вокруг оси Ox. Вначале изобразим тело.

$$\boldsymbol{a = 1;\ xc[t\_] = a(t - Sin[t]);\ yc[t\_] = a(1 - Cos[t]);}$$
$$\boldsymbol{RevolutionPlot3D[\{xc[u], yc[u]\}, \{u, 0, 2\pi\}, RevolutionAxis \to \{1, 0, 0\}]}$$



Используем первую форму выражения (12)

```
a =.; x[t_] = a(t - Sin[t]); y[t_] = a(1 - Cos[t]);
V = π Integrate[y[t]^2 D[x[t], t], {t, 0, 2π}]
```

$5a^3\pi^2$

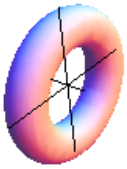
□

Пример. Определим объем тора

$$(x^2 + y^2 + z^2 + R^2 - r^2)^2 - 4R^2(y^2 + z^2) = 0$$

Изобразим тело.

```
R = 3; r = 1;
RegionPlot3D[(x^2 + y^2 + z^2 + R^2 - r^2)^2 - 4R^2(y^2 + z^2) ≤ 0,
    {x, -r, r}, {y, -(R + r), (R + r)}, {z, -(R + r), (R + r)},
    BoxRatios → {2r, 2(R + r), 2(R + r)}, Mesh → None,
    Boxed → False, AxesOrigin → {0, 0, 0}, Ticks → None]
```



Объем тора вычислим как разность объемов, образованных вращением вокруг оси *Ox* верхней и нижней полуокружностей $y = R \pm \sqrt{r^2 - x^2}$, представляющих сечение тора плоскостью *xOy* . Здесь мы воспользуемся формулой (13).

```
Clear[r, R];
V = π Integrate[(R + Sqrt[r^2 - x^2])^2 - (R - Sqrt[r^2 - x^2])^2, {x, -r, r},
    Assumptions :→ R > r > 0]
```

$2\pi^2 r^2 R$

□

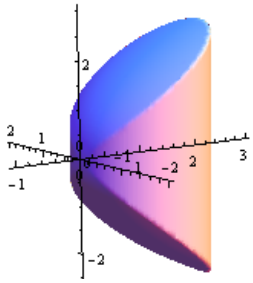
Вычисление объема с помощью тройного интеграла.

Объем трехмерного тела (V) можно вычислять как интеграл по области

$$V = \iiint_{(V)} d\,x\,d\,y\,d\,z \qquad (14)$$

Пример. Найти объем V тела, вырезанного цилиндром $x^2 + y^2 = 2\,a\,x$ из параболоида вращения $y^2 + z^2 = 4a\,x$. Вначале изобразим тело (V).

```
a = 1;
RegionPlot3D[x^2 + y^2 ≤ 2ax && y^2 + z^2 ≤ 4ax,
    {x, -1, 3}, {y, -2, 2}, {z, -3, 3}, PlotPoints → 100,
    Boxed → False, AxesOrigin → {0, 0, 0}, Mesh → None]
```



Теперь находим объем V.

$\boldsymbol{a} =.;$

$\mathbf{Integrate}[\mathbf{Boole}[\boldsymbol{x}^2 + \boldsymbol{y}^2 \leq 2\,\boldsymbol{a}\,\boldsymbol{x}\ \&\&\ \boldsymbol{y}^2 + \boldsymbol{z}^2 \leq 4\,\boldsymbol{a}\,\boldsymbol{x}],$
$\{\boldsymbol{x}, -\infty, \infty\}, \{\boldsymbol{y}, -\infty, \infty\}, \{\boldsymbol{z}, -\infty, \infty\}, \mathbf{Assumptions} :\to \boldsymbol{a} > 0]$

$\frac{2}{3}a^3(8 + 3\pi)$

□

Вычисление объема с помощью двойного интеграла.

Объем вертикального цилиндрического тела, имеющего своим основанием область D на плоскости xOy и ограниченного сверху поверхностью $z = f(x, y)$, выражается двойным интегралом

$$V = \iint\limits_D z\,d\,x\,d\,y = \iint\limits_D f(x, y)\,d\,x\,d\,y \qquad (15)$$

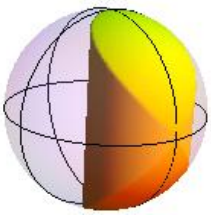
Пример. Найти объем V тела, вырезанного цилиндром $x^2 + y^2 = R\,x$ из сферы $x^2 + y^2 + z^2 = R^2$. Выполним вычисление по формуле (15). Имеем

$\boldsymbol{R} =.;\ \ \boldsymbol{z}[\mathbf{x\_}, \mathbf{y\_}] = \sqrt{\boldsymbol{R}^2 - \boldsymbol{x}^2 - \boldsymbol{y}^2};$
$\boldsymbol{V} = 2\,\mathbf{Integrate}[\boldsymbol{z}[\boldsymbol{x}, \boldsymbol{y}]\mathbf{Boole}[\boldsymbol{x}^2 + \boldsymbol{y}^2 \leq \boldsymbol{R}\boldsymbol{x}], \{\boldsymbol{x}, -\infty, \infty\}, \{\boldsymbol{y}, -\infty, \infty\},$
$\mathbf{Assumptions} :\to \boldsymbol{R} > 0]$

$\frac{2}{9}(-4 + 3\pi)R^3$

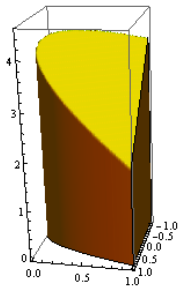
Теперь изобразим тело (V).

$\boldsymbol{R} = \mathbf{1};$
$\mathbf{p1} = \mathbf{RegionPlot3D}[\boldsymbol{x}^2 + \boldsymbol{y}^2 \leq \boldsymbol{R}\,\boldsymbol{x}\ \&\&\ \boldsymbol{x}^2 + \boldsymbol{y}^2 + \boldsymbol{z}^2 \leq \boldsymbol{R}^2,$
$\{\boldsymbol{x}, -\boldsymbol{R}, \boldsymbol{R}\}, \{\boldsymbol{y}, -\boldsymbol{R}, \boldsymbol{R}\}, \{\boldsymbol{z}, -\boldsymbol{R}, \boldsymbol{R}\}, \mathbf{PlotPoints} \to \mathbf{100},$
$\mathbf{Mesh} \to \mathbf{None}, \mathbf{PlotStyle} \to \mathbf{Yellow}];$
$\mathbf{p2} = \mathbf{RegionPlot3D}[\boldsymbol{x}^2 + \boldsymbol{y}^2 + \boldsymbol{z}^2 \leq \boldsymbol{R}^2,$
$\{\boldsymbol{x}, -\boldsymbol{R}, \boldsymbol{R}\}, \{\boldsymbol{y}, -\boldsymbol{R}, \boldsymbol{R}\}, \{\boldsymbol{z}, -\boldsymbol{R}, \boldsymbol{R}\}, \mathbf{PlotPoints} \to \mathbf{50},$
$\mathbf{Mesh} \to \{\mathbf{1}, \mathbf{1}, \mathbf{1}\}, \mathbf{PlotStyle} \to \mathbf{Opacity}[\mathbf{0.2}]];$
$\mathbf{Show}[\mathbf{p2}, \mathbf{p1}, \mathbf{Boxed} \to \mathbf{False}, \mathbf{Axes} \to \mathbf{None}, \mathbf{BoxRatios} -> \{\mathbf{1}, \mathbf{1}, \mathbf{1}\}]$



Пример. Найти объем V вертикального цилиндра, который сверху ограничен частью плоскости $x+y+z=4$, снизу – частью плоскости *xOy*, заключенной между параболой $y=x^2$ и прямой $y=1$. Сперва изобразим тело.

```
RegionPlot3D[x + y + z ≤ 4 && y ≥ x^2 && y ≤ 1,
  {x, -1, 1}, {y, 0, 1}, {z, 0, 4.5}, PlotPoints → 100,
  Mesh → None, PlotStyle → Yellow, BoxRatios → {2, 2, 4.5}]
```



Теперь вычисляем объем.

```
f[x_, y_] = 4 - x - y;
V = Integrate[f[x, y] Boole[x^2 ≤ y ≤ 1], {x, -1, 1}, {y, 0, 1}]
```

$$\frac{68}{15}$$

Площадь поверхности тела.

Площадь поверхности, заданной явным уравнением.

Пусть поверхность задана явным уравнение $z=f(x,y)$. Площадь куска этой поверхности, который проектируется на плоскость *xOy* в виде замкнутой области *D*, вычисляется по формуле

$$S=\iint_D \sqrt{1+f_x^2+f_y^2}\,dx\,dy \qquad (16)$$

Пример. Вычислим площадь поверхности сферы. Уравнение $z=\sqrt{R^2-x^2-y^2}$ представляет полусферу радиуса R. Учитывая, что (16) даст площадь только верхней полусферы, имеем

```
z = Sqrt[R^2 - x^2 - y^2];
ds = Simplify[Sqrt[1 + D[z, x]^2 + D[z, y]^2], Assumptions :→ R > 0]
S = 2 Integrate[ds Boole[x^2 + y^2 ≤ R^2], {x, -∞, ∞}, {y, -∞, ∞},
                                          Assumptions :→ R > 0]
```

$4\pi R^2$

Это совпадает с известной формулой элементарной геометрии.

□

Пример. Вычислим площадь среза кругового цилиндра $x^2+y^2\le R^2$ плоскостью $a\,x+b\,y+c\,z+d=0$

```
a = 1; b = 1; c = 1; d = -3; R = 1;
RegionPlot3D[x^2 + y^2 ≤ R^2 && z ≥ 0 && a x + b y + c z + d ≤ 0,
  {x, -R, R}, {y, -R, R}, {z, 0, 5}, PlotPoints → 100,
  BoxRatios → {2R, 2R, 5}, Mesh → None, Boxed → False,
  AxesOrigin → {0, 0, 0}, Ticks → None]
```

```
Clear[x, y, z, R, a, b, c, d];
z = (d − a x − b y)/c;
ds = √(1 + D[z, x]^2 + D[z, y]^2)
S = Integrate[ds Boole[x^2 + y^2 ≤ R^2], {x, −∞, ∞}, {y, −∞, ∞},
              Assumptions :→ R > 0&&c > 0]
```

$$\frac{\sqrt{a^2+b^2+c^2}\,\pi R^2}{c}$$

□

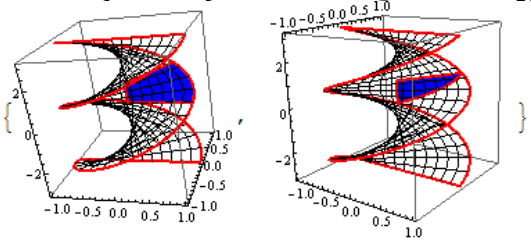
Если поверхность задана в векторном виде $\mathbf{r}=\mathbf{r}(u,v)$, то площадь участка поверхности вычисляется по формуле

$$S=\iint_D \sqrt{EG-F^2}\,du\,dv, \qquad (17)$$

где D – область в плоскости изменения параметров u,v, а величины $E=\mathbf{r}_u^2$, $F=\mathbf{r}_u\,\mathbf{r}_v$, $G=\mathbf{r}_v^2$ являются коэффициентами первой квадратичной формы поверхности.

Пример. На прямом геликоиде $x=u\cos v$, $y=u\sin v$, $z=av$ найти площадь четырехугольника, ограниченного линиями $u=0, u=a, v=0, v=1$, а также площадь криволинейного треугольника $0\le u\le av$, $0\le v\le 1$. Вначале изобразим эти области.

```
Clear[r, u, v];
a = 1; r = {u Cos[v], u Sin[v], a v};
p1 = ParametricPlot3D[r, {u, −1, 1}, {v, −π, π}, PlotStyle → None,
                      Mesh → {11, 21}, BoundaryStyle → {Red, Thick}];
p2 = ParametricPlot3D[r, {u, 0, a}, {v, 0, 1}, Mesh → None,
                      PlotStyle → Blue, BoundaryStyle → {Red, Thick}];
p3 = ParametricPlot3D[r, {v, 0, 1}, {u, 0, av}, Mesh → None,
                      PlotStyle → Blue, BoundaryStyle → {Red, Thick}];
{Show[p1, p2, BoxRatios → {1, 1, 1}, PlotRange → All],
 Show[p3, p1, BoxRatios → {1, 1, 1}, PlotRange → All]}
```



Теперь вычисляем площади областей.

```
Clear[r, ru, rv, a];
r = {u Cos[v], u Sin[v], a v};
ru = D[r, u]; rv = D[r, v];
g11 = Simplify[ru.ru];
g12 = Simplify[ru.rv];
g22 = Simplify[rv.rv];
ds = Simplify[Sqrt[g11g22 - g12^2], Assumptions :> a > 0];
S1 = Integrate[ds, {u, 0, a}, {u, 0, 1}, Assumptions :> a > 0]
S2 = Integrate[ds, {v, 0, 1}, {u, 0, av}, Assumptions :> a > 0];
Simplify[S2 /. ArcSinh[x_] → Log[x + Sqrt[x^2 + 1]]]
```

$$\frac{1}{2}a\left(\sqrt{1+a^2}+a^2\,\mathrm{Log}\left[\frac{1+\sqrt{1+a^2}}{a}\right]\right)$$

$$-\frac{1}{6}a^2\left(-2+\sqrt{2}-3\,\mathrm{Log}[1+\sqrt{2}]\right)$$

При вычислении площади S_2 криволинейного треугольника мы предпочли сделать замену $\operatorname{arcsinh} x = \ln\left(x+\sqrt{x^2+1}\right)$.

Дадим пояснения к подстановке ArcSinh[x_] → Log[$x + \sqrt{x^2+1}$]. Когда в выражении `expr` выполняется подстановка `expr/.f[x]→g[x]`, то `f[x]` означает, что функция `f` вычисляется при некотором конкретном значении аргумента x и, если в `expr` находится точное соответствие `f[x]`, то оно заменяется на `g[x]`. Выражения `f` с другими аргументами не заменяются! Подстановка `f[x_]→g[x]` соответствует случаю произвольного аргумента у функции `f`. Поэтому применение правила замены `f[x]→g[x]` и `f[x_]→g[x]` к выражению `f[x]+f[y]+f[z]` приводит к разным результатам.

```
f[x] + f[y] + f[z] /. f[x] → g[x]
```

$f[y] + f[z] + g[x]$

```
f[x] + f[y] + f[z] /. f[x_] → g[x]
```

$g[x] + g[y] + g[z]$

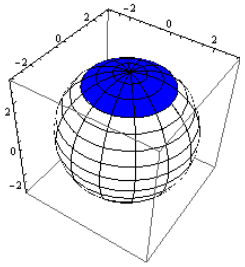
В коде, приведенном выше, для замены мы использовали подстановку с шаблоном `f[x_]→g[x]`, где в качестве функции `f` выступал арксинус гиперболический, а в качестве `g[x]` – выражение Log[$x + \sqrt{x^2+1}$] .

□

Пример. На сфере $\boldsymbol{r} = \left\{R\,\mathrm{Sin}\left[\frac{u}{R}\right]\mathrm{Cos}[v],\ R\mathrm{Sin}\left[\frac{u}{R}\right]\mathrm{Sin}[v],\ R\mathrm{Cos}\left[\frac{u}{R}\right]\right\}$ ($0 \le u \le \pi R$ и $0 \le v < 2\pi$) вычислить площадь круга радиуса r_0 (т.е. $0 \le u \le r_0$). Вначале нарисуем область «круга» на сфере.

```
Remove[r]; R = 3; r0 = 2;
r = {R Sin[u/R] Cos[v], R Sin[u/R] Sin[v], R Cos[u/R]};
sph = ParametricPlot3D[r, {u, 0, πR}, {v, 0, 2π}, Mesh → {11, 13, 0},
        Lighting → {{Ambient, White}}, BoundaryStyle → Black,
        RegionFunction → ((#4 > r0)&)];
```

```
crc = ParametricPlot3D[r, {u, 0, r0}, {v, 0, 2π}, Mesh → {2, 13, 0},
                PlotStyle → Blue];
Show[sph, crc, PlotRange → All]
```



Обратите внимание, что для наложения «круга» на сферу мы из нее исключили область этого «круга».

Теперь вычисляем площадь «круга» по формуле (17).

```
Clear[R, r, ru, rv, r0];
r = {R Sin[u/R] Cos[v], R Sin[u/R] Sin[v], R Cos[u/R]};
ru = D[r, u]; rv = D[r, v];
g11 = Simplify[ru.ru];
g12 = Simplify[ru.rv];
g22 = Simplify[rv.rv];
ds = Simplify[√(g11g22 − g12^2), Assumptions :→ R > 0 && u > 0];
S = Integrate[ds, {u, 0, r0}, {v, 0, 2π}, Assumptions :→ R > 0 && 0 < r0 < R];
TrigFactor[S]
```

$4\pi R^2 \mathrm{Sin}[\frac{\mathrm{r0}}{2R}]^2$

Если $r_0 = \pi R$, то кругом радиуса r_0 на сфере является сфера. Ее площадь будет $S = 4\pi R^2 \sin\frac{\pi R}{2R} = 4\pi R^2$, что совпадает с классической формулой.

□

Площадь поверхности, получаемой вращением кривой $y = f(x)\ (a \le x \le b)$ *вокруг оси* Ox, вычисляется по формуле

$$S = 2\pi \int_a^b y\sqrt{1 + y'^2_x}\,dx = 2\pi \int_a^b f(x)\sqrt{1 + (f'(x))^2}\,dx \qquad (18)$$

Пример. Вычислить площадь поверхности шарового пояса. Пусть полуокружность, описанная около начала радиусом r, вращается вокруг оси *x*. Тогда $y = \sqrt{r^2 - x^2}$. Вычисляем

```
r =.; f[x_] = √(r^2 − x^2);
sr = Simplify[f[x]√(1 + D[f[x], x]^2), Assumptions :→ r > 0 && 0 ≤ x < r]
S = 2π ∫_x1^x2 sr dx
S/.(x2 − x1) → h
```

$2\,\pi\,r(-\mathrm{x1} + \mathrm{x2})$

$2\,h\,\pi\,r$

Таким образом, площадь поверхности пояса, описанного дугой, концы которой имеют абсциссы x_1 и x_2, будет равна $2\pi r h$, где h есть высота пояса. В частности при $x_1 = -r$, $x_2 = r$ получаем площадь всей сферы $4\pi r^2$.

□

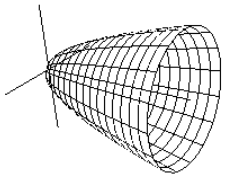
Пример. Вычислить площадь поверхности, образованной вращением вокруг оси Ox части кривой $y^2 = 4 + x$, отсеченной прямой x=2.

Уравнение $y^2 = 4 + x$ определяет параболу с вершиной в точке (-4, 0) и осью симметрии Ox, поэтому для вычисления площади поверхности вращения достаточно рассмотреть одну ветвь параболы $y = \sqrt{4 + x}$ на отрезке [-4,2].

```
f[x_] = Sqrt[4 + x];
sr = Simplify[f[x] Sqrt[1 + D[f[x], x]^2], Assumptions :> x > -4]
S = 2 Pi Integrate[sr, {x, -4, 2}]
```

$$\frac{62\pi}{3}$$

```
RevolutionPlot3D[f[x], {x, -4, 2}, RevolutionAxis -> {1, 0, 0},
    Boxed -> False, AxesOrigin -> {-4, 0, 0},
    Lighting -> {{"Ambient", White}}, Ticks -> None]
```



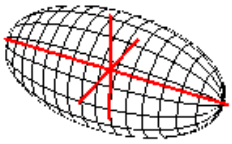
Пример. Определим площадь поверхности эллипсоида вращения. Если эллипс $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$ вращается вокруг оси x, то для вычисления площади поверхности вращения достаточно рассмотреть одну ветвь эллипса $y = b\sqrt{1 - x^2/a^2}$. Будем также полагать $a > b$. Тогда имеем

```
Clear[a, b];
f[x_] = b Sqrt[1 - x^2/a^2];
sr = Simplify[f[x] Sqrt[1 + D[f[x], x]^2],
      Assumptions :> a > 0 && b > 0 && -a < x < a];
S = 2 Pi Integrate[sr, {x, -a, a}, Assumptions :> 0 < b < a && a > b];
S /. (a^2 - b^2)^q_ -> (a p)^(2 q)
```

$$2\,b\,\pi\left(b + \frac{a\,\mathrm{ArcSin}[p]}{p}\right)$$

Здесь $p = \frac{\sqrt{a^2 - b^2}}{a}$ является эксцентриситетом эллипса.

```
a = 2; b = 1;
RevolutionPlot3D[f[x], {x, -a, a}, RevolutionAxis → {1, 0, 0},
                 Boxed → False, AxesOrigin → {0, 0, 0},
                 Lighting → {{Ambient, White}}, Ticks → False,
                 AxesStyle → Directive[Red, Thick]]
```



Если *кривая задана параметрическими уравнениями* $x = x(t),\ y = y(t)\ \ (t_0 \le t \le T)$, то *площадь поверхности вращения вокруг оси* x вычисляется по формуле

$$S = 2\pi \int_{t_0}^{T} y(t)\sqrt{(x'(t))^2 + (y'(t))^2}\, dt \qquad (19)$$

Пример. Вычислим площадь поверхности тора с расстоянием от центра образующей окружности до оси вращения R и с радиусом образующей окружности r $(r < R)$. Эту площадь вычислим как площадь поверхности, образуемой вращением окружности $x = r\cos t,\ \ y = R + r\sin t$ вокруг оси Ox.

```
x[t_] = rCos[t];
y[t_] = R + rSin[t];
ds = Simplify[y[t]√(D[x[t], t]^2 + D[y[t], t]^2],
                Assumptions :→ R > 0 && 0 < r < R]
S = 2 π Integrate[ds, {t, 0, 2π}, Assumptions :→ 0 < r < R && R > 0]
```

$4\pi^2 rR$

Это можно представить как $S = 2\pi r \cdot 2\pi R$. Стало быть, площадь тора равна произведению длины образующей окружности на длину пути, описываемого ее центром.

□

Пример. Дана циклоида $x = a(t - \sin t),\ \ y = a(1 - \cos t)$. Найти площадь поверхности, образованной вращением ее вокруг оси Ox.

```
a =.; x[t_] = a(t - Sin[t]); y[t_] = a(1 - Cos[t]);
ds = Simplify[y[t]√(D[x[t], t]^2 + D[y[t], t]^2], Assumptions :→ a > 0]
S = 2 π Integrate[ds, {t, 0, 2π}, Assumptions :→ a > 0]
```

$$\frac{64a^2\pi}{3}$$

□

Для *полярной системы координат*, если вращение кривой $\rho = \rho(\varphi)$ производится вокруг полярной оси, *площадь поверхности тела вращения* вычисляется по формуле

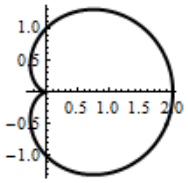
$$S = 2\pi \int_{\alpha}^{\beta} \rho(\varphi)\sin\varphi\sqrt{\rho^2(\varphi) + (\rho'(\varphi))^2}\, d\varphi \qquad (20)$$

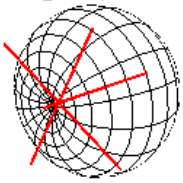
Пример. Найти площадь поверхности, образованной вращением кардиоиды $r=a(1+\cos\varphi)$ вокруг полярной оси. Поскольку кардиоида симметрична относительно полярной оси, то пределами интегрирования в формуле (20) нужно выбрать 0 и π. Имеем

```
a =.; r[φ_] = a(1 + Cos[φ]);
ds = Simplify[r[t]Sin[t]√(r[t]^2 + D[r[t], t]^2), Assumptions :→ a > 0];
S = 2 π Integrate[ds, {t, 0, π}, Assumptions :→ a > 0]
```

$$\frac{32a^2\pi}{5}$$

```
a = 1;
PolarPlot[r[φ], {φ, 0, 2π}, PlotStyle → {Black, Thick}] (* рис. слева *)
RevolutionPlot3D[{r[t]Cos[t], r[t]Sin[t]}, {t, 0, π},
    RevolutionAxis → {1, 0, 0}, Boxed → False, AxesOrigin → {0, 0, 0},
    Lighting → {{Ambient, White}}, Ticks → False,
    AxesStyle → Directive[Red, Thick]]   (* рисунок справа *)
```





3.4 Векторный анализ

Векторный анализ изучает скалярные и векторные поля. Говорят, что в некоторой области задано поле, если каждой точке этой области соответствует определенное значение некоторой величины – числовой или векторной.

Если в каждой точке области задана величина, принимающая числовые значения, то поле называется скалярным, если же в каждой точке области задан вектор, то поле называется векторным.

Если поле является скалярным, то задание поля равносильно заданию функции трех переменных $u(x,y,z)$ (или двух, если поле плоское). Если поле векторное, то надо задать все проекции переменного вектора на оси координат $\vec{\mathbf{A}} = P(x,y,z)\mathbf{i} + Q(x,y,z)\mathbf{j} + R(x,y,z)\mathbf{k}$, где P, Q, R – скалярные функции.

Векторный анализ является основой многих физических и математических моделей и, естественно, что в системе *Mathematica* имеются функции, выполняющие основные операции векторного анализа. При этом понятия «вектор» в *Mathematica* нет. Вместо этого используется понятие списка, который может быть одномерным (аналог вектора), двумерным (аналог матрицы) и многомерным. Для многих функций, рассматриваемых в данном разделе, аргументами могут быть многомерные списки. Однако мы уделим внимание в основном одномерным спискам, представляющим вектора и вектор – функции.

3.4.1 Алгебраические операции с векторами

Поскольку векторный анализ имеет дело с семействами функций (в терминологии *Mathematica* со списками функций), то уместно напомнить о некоторых функциях, выполняющих алгебраические операции с векторами/списками.

Функция `Norm[expr]` возвращает норму числа, вектора или матрицы.

Norm[{x, y, z}]

$\sqrt{\text{Abs}[x]^2 + \text{Abs}[y]^2 + \text{Abs}[z]^2}$

или

Simplify[**Norm**[{x, y, z}],
Assumptions :→ $x \in$ **Reals**&&$y \in$ **Reals**&&$z \in$ **Reals**]

$\sqrt{x^2 + y^2 + z^2}$

Norm[$-3 + 4I$]

5

Norm[{{1, 2}, {−3, 4}}]

$\sqrt{5(3 + \sqrt{5})}$

В формате `Norm[{x1,x2,...},n]` вычисляется выражение $\left(x_1^n + x_2^n + \ldots\right)^{1/n}$. Если $n = \infty$, то вычисляется норма $\max\left(|x_1|, |x_2|, \ldots\right)$.

Norm[{x, y, z}, n]

$(\text{Abs}[x]^n + \text{Abs}[y]^n + \text{Abs}[z]^n)^{\frac{1}{n}}$

Norm[{1, 2, 3}, 4]

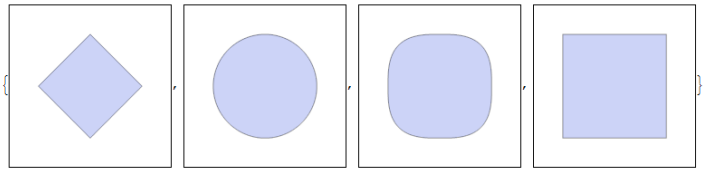
$2^{1/4}\sqrt{7}$

Norm[{x, y, z}, **Infinity**]

Max[Abs[x], Abs[y], Abs[z]]

Вот графический пример, иллюстрирующий применение функции `Norm`.

Table[**RegionPlot**[**Norm**[{x, y}, p] ≤ 1, {x, −1.5, 1.5}, {y, −1.5, 1.5},
FrameTicks → **None**], {p, {1, 2, 3, **Infinity**}}]



Функция `Normalize[список]` возвращает нормированный вектор, координаты которого изначально были заданы `списком`.

Normalize[{2, 3, 6}]

$\{\frac{2}{7}, \frac{3}{7}, \frac{6}{7}\}$

Simplify[**Normalize**[{$u + v, u - v$}],
Assumptions :→ $u \in$ **Reals** && $v \in$ **Reals**]

$\{\frac{u + v}{\sqrt{2}\sqrt{u^2 + v^2}}, \frac{u - v}{\sqrt{2}\sqrt{u^2 + v^2}}\}$

$v = \{1, 2 + 3I, 4I, 5 + 6I\};$

Normalize[v]

$\{\frac{1}{\sqrt{91}}, \frac{2+3i}{\sqrt{91}}, \frac{4i}{\sqrt{91}}, \frac{5+6i}{\sqrt{91}}\}$

Функция `Orthogonalize[{v1,v2,...}]` строит нормированный ортогональный базис из векторов `v1,v2,...`.

Orthogonalize[{{**2**, **0**, **0**}, {**2**, **1**, **0**}, {**3**, **2**, **1**}}]

```
{{1,0,0},{0,1,0},{0,0,1}}
```

Те же вектора, но в другой последовательности.

Orthogonalize[{{**2**, **1**, **0**}, {**3**, **2**, **1**}, {**2**, **0**, **0**}}]

$\{\{\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}}, 0\}, \{-\frac{1}{\sqrt{30}}, \sqrt{2/15}, \sqrt{5/6}\}, \{\frac{1}{\sqrt{6}}, -\sqrt{2/3}, \frac{1}{\sqrt{6}}\}\}$

Если вектор в списке один, то он просто нормируется.

Orthogonalize[{{**3**, **1**, **1**}}]

$\{\{\frac{3}{\sqrt{11}}, \frac{1}{\sqrt{11}}, \frac{1}{\sqrt{11}}\}\}$

Если векторов меньше, чем размерность пространства, то выполняется обычная ортогонализация

Orthogonalize[{{**1**, −**1**, **1**}, {**0**, −**1**, −**1**}}]

$\{\{\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\}, \{0, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\}\}$

Если векторов в списке больше, чем размерность пространства, то последние вектора возвращаются нулевыми.

Orthogonalize[{{**1**, −**1**, **1**}, {**0**, −**1**, −**1**}, {**0**, **3**, **1**}, {−**1**, **0**, **0**}}]

$\{\{\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\}, \{0, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\}, \{\sqrt{\frac{2}{3}}, \frac{1}{\sqrt{6}}, -\frac{1}{\sqrt{6}}\}, \{0,0,0\}\}$

Функцию `Orthogonalize` можно применять к матрицам (спискам списков)

Simplify[**Orthogonalize**[{{x, −y}, {$x+y$, $x-y$}}],
Assumptions ⩴ (x ∈ **Reals**&&y ∈ **Reals**)]

$\{\{\frac{x}{\sqrt{x^2+y^2}}, -\frac{y}{\sqrt{x^2+y^2}}\}, \{\frac{y}{\sqrt{x^2+y^2}}, \frac{x}{\sqrt{x^2+y^2}}\}\}$

Функция `Orthogonalize` может использовать свою собственную функцию скалярного произведения, может обрабатывать многомерные списки, имеет две опции. Подробнее с ней вы можете познакомиться по справочной системе.

Функция `Dot` вычисляет скалярное произведение векторов, а также может выполнить матричное умножение списков (матриц) и умножение матрицы на вектор

Dot[{x, y, z}, {a, b, c}]

$ax + by + cz$

Инфиксной формой функции `Dot` является оператор '**.**' (точка). С его помощью мы можем вычислить произведение векторов и/или матриц.

{{a,b},{c,d}}.{x,y}

```
{ax+by,cx+dy}
```

или

```
{x,y}.{{a,b},{c,d}}
{ax+cy,bx+dy}
```

В первом случае `{x,y}` выступает, как вектор-столбец, во втором - как вектор-строка. В следующем выражении присутствуют оба типа.

$$\{x, y\}.\{x, y\}$$

$$x^2 + y^2$$

Векторное произведение векторов выполняется функцией `Cross`.

```
Cross[{1,2,3},{2,4,5}]
{-2,1,0}
```

Инфиксной формой функции `Cross` является оператор '×' (крест). Он вводится как `Esc - cross - Esc`.

$$\{x, y, z\} \times \{a, b, c\}$$

$$\{cy - bz, -cx + az, bx - ay\}$$

Функция `EuclideanDistance[u,v]` вычисляет евклидово расстояние между векторами

$$\mathbf{EuclideanDistance}[\{a, b, c\}, \{x, y, z\}]$$

$$\sqrt{\text{Abs}[a - x]^2 + \text{Abs}[b - y]^2 + \text{Abs}[c - z]^2}$$

Если вы уверены, что координаты векторов вещественные, то можно избавиться от модулей, выполнив замену

$$\mathbf{EuclideanDistance}[\{a, b, c\}, \{x, y, z\}] /. (\mathbf{Abs[x\_]\hat{}2} \to \boldsymbol{x\hat{}2})$$

$$\sqrt{(a - x)^2 + (b - y)^2 + (c - z)^2}$$

Здесь подстановка Abs[x_]^2 → x^2 любой «квадрат модуля выражения» заменяет «квадратом выражения», т.е. выражение вида $Abs[expr]^2$ заменяется выражением $expr^2$ при любом $expr$.

Евклидово расстояние эквивалентно норме разности векторов.

$$\mathbf{EuclideanDistance}[\{a, b, c\}, \{x, y, z\}] == \mathbf{Norm}[\{a, b, c\} - \{x, y, z\}]$$

```
True
```

Функция `VectorAngle[u,v]` вычисляет угол между векторами `u` и `v`.

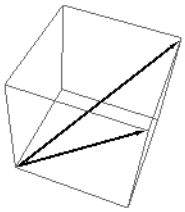
$$\mathbf{u} = \{1, 0, 1\};$$

$$\mathbf{v} = \{1, 1, 1\};$$

$$\mathbf{VectorAngle}[\mathbf{u}, \mathbf{v}]$$

$$\text{ArcCos}[\sqrt{2/3}]$$

$$\mathbf{Graphics3D}[\{\mathbf{Thick}, \mathbf{Arrow}[\{\{0, 0, 0\}, \mathbf{u}\}], \mathbf{Arrow}[\{\{0, 0, 0\}, \mathbf{v}\}]\}]$$



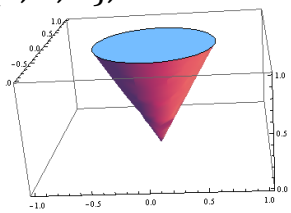
Если координаты векторов вещественные, то выполняя очевидные замены, можно получить классическое выражение для угла между векторами

VectorAngle[{a, b, c}, {x, y, z}]/.{**Conjugate**[x_] → x, **Abs**[x_]^2 → x^2}

$$\text{ArcCos}\left[\frac{a\,x + b\,y + c\,z}{\sqrt{a^2 + b^2 + c^2}\sqrt{x^2 + y^2 + z^2}}\right]$$

Вот один пример использования функции `VectorAngle`.

RegionPlot3D[**VectorAngle**[{**0**, **0**, **1**}, {x, y, z}] < **Pi/6** && $z \le 1$,
{$x, -1, 1$}, {$y, -1, 1$}, {z, **0**, **1**}, **BoxRatios** → {**2**, **2**, **1**}]



Функция `Projection[u,v]` вычисляет проекцию вектора `u` на вектор `v`.

Projection[{**4**, −**6**, **3**}, {**1**, **2**, **3**}]

$$\left\{\frac{1}{14}, \frac{1}{7}, \frac{3}{14}\right\}$$

Выражение для вычисления проекции имеет вид $\frac{\bar{\mathbf{u}} \cdot \mathbf{v}}{\bar{\mathbf{v}} \cdot \mathbf{v}}\mathbf{v}$, где черта сверху означает комплексное сопряжение. Для вещественных векторов оно совпадает с классическим определением.

Пример. Разложить вектор `u` на составляющие: вдоль вектора `v` и ортогональный к `v` вектор.

u = {**1**, **1**, **1**};

v = {**1**, **2**, **4**};

upar = **Projection**[u, v]

$$\left\{\frac{1}{3}, \frac{2}{3}, \frac{4}{3}\right\}$$

uort = u − **uparallel**

$$\left\{\frac{2}{3}, \frac{1}{3}, -\frac{1}{3}\right\}$$

Проверим, что вектора `uort` и `upar` в сумме дают вектор `u` и, что они ортогональны.

upar + **uort**

```
{1,1,1}
```

uort. **upar**

0

□

В формате `Projection[u,v,f]` вычисляется проекция вектора `u` на вектор `v`, используя в качестве скалярного произведения функцию `f`. Например, если координаты векторов вещественные, то в качестве `f` надо выбирать функцию `Dot`.

Projection[{x, y, z}, {a, b, c}, **Dot**]

$$\left\{\frac{a(ax + by + cz)}{a^2 + b^2 + c^2}, \frac{b(ax + by + cz)}{a^2 + b^2 + c^2}, \frac{c(ax + by + cz)}{a^2 + b^2 + c^2}\right\}$$

Под векторами можно понимать функции. Тогда в качестве скалярного произведения можно использовать определенный интеграл. Например

$$\mathbf{ip}[\mathbf{f1\_}, \mathbf{f2\_}] := \mathbf{Integrate}[\mathbf{f1} * \mathbf{f2}, \{x, -\pi/2, \pi/2\}]$$

$$\mathbf{Projection}[x^2, \mathbf{Cos}[x], \mathbf{ip}]$$

$$\frac{(-8 + \pi^2)\mathrm{Cos}[x]}{\pi}$$

Здесь мы создали функцию `ip[f1,f2]`, которая при вычислении проекции используется для вычисления скалярного произведения функций.

Функция `Thread[f[args]]` «проносит» `f` сквозь любой список/вектор, который появляется в аргументе `args`. В результате создается список значений функции `f` на элементах списков из `args`.

```
Thread[f[{a,b,c}]]
{f[a],f[b],f[c]}
```

$$\mathbf{Thread}[f[\{a, b, c\}, x]]$$

$$\{f[a, x], f[b, x], f[c, x]\}$$

$$\mathbf{Thread}[f[\{a, b, c\}, \{x, y, z\}]]$$

$$\{f[a, x], f[b, y], f[c, z]\}$$

```
Thread[{a,b,c}=={x,y,z}]
{a==x,b==y,c==z}
```

Последний результат станет понятен, если аргумент функции `Thread` представить в полной форме

$$\{a, b, c\} == \{x, y, z\}//\mathbf{FullForm}$$

$$\mathrm{Equal}[\mathrm{List}[a, b, c], \mathrm{List}[x, y, z]]$$

Т.о. функция `Equal` «проносится» внутрь своих списочных аргументов.

В формате `Thread[f[args],h]` функция `Thread` «проносит» `f` сквозь любой объект с заголовком `h`, который появляется в аргументе `args`.

$$\mathbf{Thread}[\mathbf{Sin}[x == y], \mathbf{Equal}]$$

$$\mathrm{Sin}[x] == \mathrm{Sin}[y]$$

Рассмотрим теперь преобразование поворота векторов. Оно выполняется функцией `RotationTransform`.

Вначале рассмотрим вектора на плоскости. В формате `RotationTransform[θ]` возвращается матрица преобразования поворота, которую можно применить к любой паре чисел (двумерному вектору или координатам точки на плоскости). Следует уточнить, что возвращается объект – функция `TransformationFunction`, который действует как функция и который можно применять к одномерным спискам.

$$\mathbf{RotationTransform}[\theta]$$

$$\mathrm{TransformationFunction}\left[\left(\begin{array}{cc|c} \mathrm{Cos}[\theta] & -\mathrm{Sin}[\theta] & 0 \\ \mathrm{Sin}[\theta] & \mathrm{Cos}[\theta] & 0 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$$

Если такую функцию применить к вектору, то мы получим повернутый вектор.

$$\mathbf{RotationTransform}[\pi/4][\{1, 1\}]$$

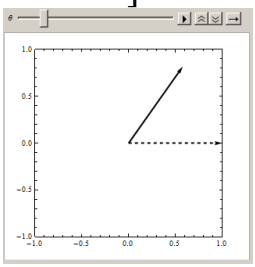
$$\{0, \sqrt{2}\}$$

$\mathbf{RotationTransform}[\boldsymbol{\theta}][\{\boldsymbol{x},\boldsymbol{y}\}]$

$\{x\mathrm{Cos}[\theta]-y\mathrm{Sin}[\theta],y\mathrm{Cos}[\theta]+x\mathrm{Sin}[\theta]\}$

В следующем коде мы создаем анимацию вращения вектора. Меняя угол поворота от 0 до $2\,\pi$, мы пересчитываем координаты его конца с помощью функции `RotationTransform` для каждого нового значения θ .

```
pnt = {1, 0};
gpstart = Graphics[{Thick, Dashed, Arrow[{{0, 0}, pnt}]}];
Animate[
   pr = RotationTransform[θ][pnt];
   gpr = Graphics[{Thick, Arrow[{{0, 0}, pr}]}];
   Show[gpstart, gpr, Frame → True, PlotRange → {{-1, 1}, {-1, 1}}],
{θ, 0, 2π}, AnimationRunning → False]
```



В формате `RotationTransform[θ,pnt]` возвращается матрица поворота вокруг точки `pnt`. Следующая команда вычисляет координаты точки $\{x,y\}$ после поворота ее вокруг точки $\{1,0\}$ на угол θ.

$\mathbf{RotationTransform}[\boldsymbol{\theta},\{\mathbf{1},\mathbf{0}\}][\{\boldsymbol{x},\boldsymbol{y}\}]$

$\{1-\mathrm{Cos}[\theta]+x\mathrm{Cos}[\theta]-y\mathrm{Sin}[\theta],y\mathrm{Cos}[\theta]-\mathrm{Sin}[\theta]+x\mathrm{Sin}[\theta]\}$

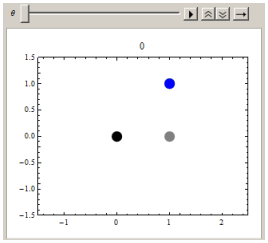
Сравните с результатом поворота вокруг начала координат.

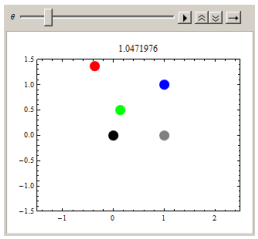
$\mathbf{RotationTransform}[\boldsymbol{\theta},\{\mathbf{0},\mathbf{0}\}][\{\boldsymbol{x},\boldsymbol{y}\}]$

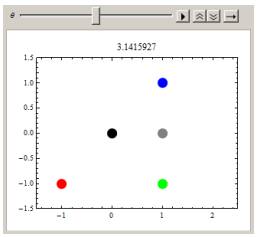
$\{x\mathrm{Cos}[\theta]-y\mathrm{Sin}[\theta],y\mathrm{Cos}[\theta]+x\mathrm{Sin}[\theta]\}$

Построим анимацию вращения точки $\{1,1\}$ одновременно вокруг двух точек: начала координат $\{0,0\}$ и точки $\{1,0\}$.

```
psize = 0.05; v = {1, 1}; pnt = {1, 0};
gpnt0 = Graphics[{Black, PointSize[psize], Point[{0, 0}]}];
gpnt1 = Graphics[{Gray, PointSize[psize], Point[{1, 0}]}];
gpntv = Graphics[{Blue, PointSize[psize], Point[v]}];
Animate[
   vr = RotationTransform[θ][v];
   gpntvr = Graphics[{Red, PointSize[psize], Point[vr]}];
   vp = RotationTransform[θ, pnt][v];
   gpntvp = Graphics[{Green, PointSize[psize], Point[vp]}];
   Show[gpnt0, gpnt1, gpntvr, gpntvp, gpntv, Frame → True,
        PlotRange → {{-1.5, 2.5}, {-1.5, 1.5}}, PlotLabel → θ],
{θ, Table[t, {t, 0, 2π, π/24}]}, AnimationRunning → False]
```







Здесь на рисунке слева показана панель анимации в нулевой момент времени. Черная точка является началом координат, вокруг которой выполняется одно вращение. Серая точка {1,0} является центром второго вращения. Синяя точка {1,1} является стартовой для обоих вращений. На среднем рисунке показано положение точек после поворота на угол $\pi/3$. Красная точка вращается вокруг начала координат, а зеленая – вокруг точки {1,0}. На правом рисунке показано положение точек после поворота на угол π.

В формате `RotationTransform[θ,u0][u]` выполняется поворот 3D вектора `u` вокруг 3D вектора u_0 на угол θ.

RotationTransform$[\boldsymbol{\theta},\{\mathbf{0},\mathbf{0},\mathbf{1}\}][\{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\}]$

$\{x\mathrm{Cos}[\theta]-y\mathrm{Sin}[\theta],y\mathrm{Cos}[\theta]+x\mathrm{Sin}[\theta],z\}$

У функции `RotationTransform` имеется еще несколько форматов вызова, с которыми вы можете познакомиться по справочной системе.

Много других типов операций со списками описаны нами в первой части пособия.

3.4.2 Дифференциальные операции векторного анализа

Градиент. Функция `Grad[f,{x1,x2,...,xn}]` вычисляет градиент $\left(\frac{\partial f}{\partial x_1},\frac{\partial f}{\partial x_2},...,\frac{\partial f}{\partial x_n}\right)$. Первым аргументом должно быть выражение `f[x1,x2,...,xn]`, а вторым – список переменных.

Grad$[\boldsymbol{f}[\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}],\{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\}]$

$\{f^{(1,0,0)}[x,y,z],f^{(0,1,0)}[x,y,z],f^{(0,0,1)}[x,y,z]\}$

Grad$[\boldsymbol{x}^{\mathbf{2}}+\boldsymbol{y}^{\mathbf{2}}+\boldsymbol{z}^{\mathbf{2}},\{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\}]$

$\{2x,2y,2z\}$

Можно вычислять градиент функции любого количества переменных (меньше и больше трех).

Grad$[\boldsymbol{f}[\boldsymbol{x},\boldsymbol{y}],\{\boldsymbol{x},\boldsymbol{y}\}]$

$\{f^{(1,0)}[x,y],f^{(0,1)}[x,y]\}$

Grad$[\boldsymbol{f}[\boldsymbol{x}],\{\boldsymbol{x}\}]$

$\{f'[x]\}$

Градиент можно вычислять в различных системах координат. Для этого имеется следующий формат `Grad[f,{x1,x2,...,xn},"система"]`, где третьим аргументом передается строка – имя системы координат.

Grad$[\boldsymbol{f}[\boldsymbol{r},\boldsymbol{\varphi},\boldsymbol{z}],\{\boldsymbol{r},\boldsymbol{\varphi},\boldsymbol{z}\},$ **"Cylindrical"**$]$

$$\{f^{(1,0,0)}[r,\varphi,z],\frac{f^{(0,1,0)}[r,\varphi,z]}{r},f^{(0,0,1)}[r,\varphi,z]\}$$

Первым аргументом может быть список функций. Например, можно вычислить градиент от градиента.

$\boldsymbol{u = \mathbf{Grad}[x^3 + y^3 + 3xy^2, \{x, y\}]}$

$\boldsymbol{\mathbf{Grad}[u, \{x, y\}]//\mathbf{MatrixForm}}$

$\{3x^2 + 3y^2, 6xy + 3y^2\}$

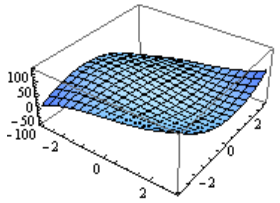
$\begin{pmatrix} 6x + 4y & 4x + 6y \\ 4x + 6y & 6x + 6y \end{pmatrix}$

Функцию `Grad` можно использовать для определения экстремальных точек функции нескольких переменных. Напомним, что если в стационарной точке функции 2 – х переменных $f'_x = 0$, $f'_y = 0$ выполняется условие $\Delta = f''_{xx} f''_{yy} - \left(f''_{xy}\right)^2 > 0$, то она является точкой строгого экстремума, а именно строгого максимума, если в ней $f''_{xx} < 0$, и строгого минимума, если $f''_{xx} > 0$.

Пример. Исследовать на экстремум функцию $f(x, y) = x^3 + 3xy^2 - 15x - 12y$.

$\boldsymbol{f[\mathrm{x\_}, \mathrm{y\_}] = x^3 + 3xy^2 - 15x - 12y;}$

$\boldsymbol{\mathbf{Plot3D}[f[x, y], \{x, -3, 3\}, \{y, -3, 3\}]}$



Вычисляем градиент функции и находим стационарные точки.

$\boldsymbol{\mathbf{grad} = \mathbf{Grad}[f[x, y], \{x, y\}]}$

$\boldsymbol{\mathbf{sol} = \mathbf{Solve}[\mathbf{grad} == \{0, 0\}, \{x, y\}]}$

$\{-15 + 3x^2 + 3y^2, -12 + 6xy\}$

$\{\{x \to -2, y \to -1\}, \{x \to -1, y \to -2\}, \{x \to 1, y \to 2\}, \{x \to 2, y \to 1\}\}$

Вычисляем матрицу Гессе $H(x) = \begin{pmatrix} f''_{xx} & f''_{xy} \\ f''_{xy} & f''_{yy} \end{pmatrix}$ функции, ее определитель (гессиан) $\Delta = f''_{xx} f''_{yy} - \left(f''_{xy}\right)^2$ и вторые производные $\left(f''_{xx}, f''_{yy}\right)$ в стационарных точках.

$\boldsymbol{\mathbf{gg} = \mathbf{Grad}[\mathbf{grad}, \{x, y\}];}$

$\boldsymbol{\mathbf{Det}[\mathbf{gg}]/.\mathbf{sol}}$

$\boldsymbol{D[f[x, y], \{x, 2\}]/.\mathbf{sol}}$

$\{108, -108, -108, 108\}$

$\{-12, -6, 6, 12\}$

Гессиан Δ положителен в первой и последней точках, следовательно они являются точками экстремума. В первой точке (-2, -1) $f''_{xx} < 0$, следовательно, это точка максимума. В точке (2, 1) $f''_{xx} > 0$ и поэтому она является точкой минимума. Другие две точки не являются точками экстремума.

Заметим, что матрицу Гессе мы вычисляли как градиент от градиента, поскольку градиент можно применять к списку функций.

```
Remove[f]
gg = Grad[Grad[f[x, y], {x, y}], {x, y}];
gg//MatrixForm
```

$$\begin{pmatrix} f^{(2,0)}[x,y] & f^{(1,1)}[x,y] \\ f^{(1,1)}[x,y] & f^{(0,2)}[x,y] \end{pmatrix}$$

□

Напомним, что в декартовой системе координат градиент можно вычислять с использованием функции `D`. В формате `D[f,{{x1,x2,…}}]` вычисляется вектор производных $\left\{\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots\right\}$ скалярной функции `f` (точнее выражения `f[x1,x2,...]`). Но этот вектор и является градиентом функции в декартовой системе.

```
D[f[x, y], {{x, y}}]
```

$\{f^{(1,0)}[x,y], f^{(0,1)}[x,y]\}$

```
D[(x^2 + y^3)^2, {{x, y}}]
```

$\{4x(x^2+y^3), 6y^2(x^2+y^3)\}$

Производная функции `u` по направлению находится как $\langle grad\, u, \vec{\mathbf{e}} \rangle$, где угловые скобки обозначают скалярное произведение векторов, а вектор $\vec{\mathbf{e}}$ является единичным вектором заданного направления.

Пример. Найти производную функции $f(x,y) = x^2 \sin 2y$ в точке $(1, \pi/2)$ в направлении вектора $\mathbf{v} = \left(\frac{3}{5}, -\frac{4}{5}\right)$. Имеем

```
f[x_, y_] = x^2 Sin[2y];
v = {3/5, -4/5};   (* единичный вектор *)
gradf = Grad[f[x, y], {x, y}]/.{x → 1, y → π/2}
```

```
{0, -2}
```

Поскольку v – единичный вектор, то производная по направлению будет

```
dirderiv = v.gradf
```

$$\frac{8}{5}$$

□

Пример. Найти производную функции $f(x,y,z) = xy + yz + xz$ в точке $(1,1,1)$ в направлении $\mathbf{v} = 2\mathbf{i} + \mathbf{j} - \mathbf{k}$. Имеем

```
F[x_, y_, z_] = xy + yz + xz;
gd = Grad[F[x, y, z], {x, y, z}]/.{x → 1, y → 1, z → 1}
```

```
{2, 2, 2}
```

```
v = {2, 1, -1};
e = Normalize[v]
```

$$\left\{\sqrt{2/3}, \frac{1}{\sqrt{6}}, -\frac{1}{\sqrt{6}}\right\}$$

dirderiv = $\boldsymbol{v}$.**gd**
4

□

Дивергенция. Функция `Div[{f1,...,fn},{x1,...,xn}]` вычисляет дивергенцию $\frac{\partial f_1}{\partial x_1}+\ldots+\frac{\partial f_n}{\partial x_n}$ вектор – функции `{f1,...,fn}`.

Div[{$\boldsymbol{f}[\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}]$, $\boldsymbol{g}[\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}]$, $\boldsymbol{h}[\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}]$},{$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$}]
$h^{(0,0,1)}[x,y,z]+g^{(0,1,0)}[x,y,z]+f^{(1,0,0)}[x,y,z]$
Div[{$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$},{$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$}]
3

Вектор – функции можно присваивать имя и использовать его вместо списка из скалярных функций.

$\boldsymbol{B}$[x_, y_, z_]:= {$\boldsymbol{x}^2,\boldsymbol{y}^2,\boldsymbol{z}^2$}
Div[$\boldsymbol{B}[\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}]$,{$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$}]
2x+2y+2z

Дивергенцию можно вычислять в криволинейных системах координат. Для этого имеется следующий формат

`Div[{f1,...,fn},{x1,...,xn},"система"],`

где третьим аргументом передается строка – имя системы координат.

Div[{$\boldsymbol{r}$**Sin**[$\boldsymbol{2\theta}$], $-\boldsymbol{r}$**Cos**[$\boldsymbol{2\theta}$]},{$\boldsymbol{r},\boldsymbol{\theta}$}, "**Polar**"]
$4\mathrm{Sin}[2\theta]$

Дивергенция вектора любой размерности является скаляром.

Div[{$\boldsymbol{w}$ **Sin**[$\boldsymbol{x}$], $\boldsymbol{w}$ **Cos**[$\boldsymbol{y}$], $\boldsymbol{x}$ $\boldsymbol{y}$ $\boldsymbol{z}$, $\boldsymbol{x}^2+\boldsymbol{w}^2$},{$\boldsymbol{w},\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$}]
$x\,z+\mathrm{Sin}[x]$

В криволинейной системе координат дивергенция вектора с постоянными компонентами может быть отлична от нуля.

Div[{**1**, **1**, **1**},{$\boldsymbol{r},\boldsymbol{\theta},\boldsymbol{\phi}$}, "**Spherical**"]//**Simplify**

$$\frac{2+\mathrm{Cot}[\theta]}{r}$$

Функцию `Div` может применяться к спискам функций.

Div[{{**f11**[$\boldsymbol{x},\boldsymbol{y}$], **f12**[$\boldsymbol{x},\boldsymbol{y}$]},{**f21**[$\boldsymbol{x},\boldsymbol{y}$], **f22**[$\boldsymbol{x},\boldsymbol{y}$]}},{$\boldsymbol{x},\boldsymbol{y}$}]
$\{\mathrm{f12}^{(0,1)}[x,y]+\mathrm{f11}^{(1,0)}[x,y],\mathrm{f22}^{(0,1)}[x,y]+\mathrm{f21}^{(1,0)}[x,y]\}$

Ротор. Функция `Curl[{f1,f2,f3},{x1,x2,x3}]` вычисляет ротор вектор – функции $(f_1(x_1,x_2,x_3),\ f_2(x_1,x_2,x_3),\ f_3(x_1,x_2,x_3))$ по формуле

$$\left(\frac{\partial f_3}{\partial x_2}-\frac{\partial f_2}{\partial x_3},\frac{\partial f_1}{\partial x_3}-\frac{\partial f_3}{\partial x_1},\frac{\partial f_2}{\partial x_1}-\frac{\partial f_1}{\partial x_2}\right).$$

Curl[{$\boldsymbol{y},-\boldsymbol{x},\boldsymbol{2z}$},{$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$}]
$\{0,0,-2\}$

Для вектор – функции двух переменных команда `Curl[{f1,f2},{x1,x2}]` вычисляет $\frac{\partial f_2}{\partial x_1}-\frac{\partial f_1}{\partial x_2}$.

$\mathbf{Curl}[\{\boldsymbol{y^2}, \boldsymbol{x^3}\}, \{\boldsymbol{x}, \boldsymbol{y}\}]$

$3x^2 - 2y$

Функция `Curl` работает даже тогда, когда ее первым аргументом является скалярная функция.

$\mathbf{Curl}[\boldsymbol{f}[\boldsymbol{x}], \{\boldsymbol{x}\}]$

$f'[x]$

$\mathbf{Curl}[\boldsymbol{f}[\boldsymbol{x}, \boldsymbol{y}], \{\boldsymbol{x}, \boldsymbol{y}\}]$

$\{-f^{(0,1)}[x, y], f^{(1,0)}[x, y]\}$

Ротор можно вычислять в криволинейных системах координат. Для этого имеется следующий формат вызова

`Curl[{f1,...,fn},{x1,...,xn},"система"],`

где третьим аргументом передается строка – имя системы координат.

$\mathbf{Curl}[\{\boldsymbol{r}\mathbf{Sin}[\mathbf{2}\boldsymbol{\theta}], -\boldsymbol{r}\mathbf{Cos}[\mathbf{2}\boldsymbol{\theta}]\}, \{\boldsymbol{r}, \boldsymbol{\theta}\}, \text{"}\mathbf{Polar}\text{"}]$

$-4\mathrm{Cos}[2\theta]$

В криволинейной системе координат ротор вектор – функции с постоянными компонентами может быть отличен от нуля.

$\mathbf{Curl}[\{\mathbf{1}, \mathbf{1}, \mathbf{1}\}, \{\boldsymbol{r}, \boldsymbol{\theta}, \boldsymbol{\phi}\}, \text{"}\mathbf{Spherical}\text{"}]$

$\left\{\frac{\mathrm{Cot}[\theta]}{r}, -\frac{1}{r}, \frac{1}{r}\right\}$

Ротор градиента равен нулю. Действительно

$\mathbf{Curl}[\mathbf{Grad}[\boldsymbol{f}[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}], \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}], \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}]$

`{0,0,0}`

Векторное поле, ротор которого тождественно равен нулю, называется безвихревым. Результат последней команды показывает, что векторное поле, являющееся градиентом некоторого скалярного поля, является безвихревым. Скалярное поле, градиентом которого является векторное поле, называется потенциалом этого векторного поля.

Например, потенциал поля точечного заряда q в пространстве задается выражением $u = -\frac{kq}{r}$, где r расстояние от точки до заряда, а k – некоторая постоянная. Действительно, градиент скалярного поля u порождает векторное поле $\boldsymbol{F}$

$\boldsymbol{u}[\mathbf{r\_}, \boldsymbol{\varphi}\mathbf{\_}, \boldsymbol{\theta}\mathbf{\_}] = -\frac{\boldsymbol{kq}}{\boldsymbol{r}};$

$\boldsymbol{F} = \mathbf{Grad}[\boldsymbol{u}[\boldsymbol{r}, \boldsymbol{\varphi}, \boldsymbol{\theta}], \{\boldsymbol{r}, \boldsymbol{\varphi}, \boldsymbol{\theta}\}, \text{"}\mathbf{Spherical}\text{"}]$

$\{\frac{kq}{r^2}, 0, 0\}$

Поскольку сила притяжения направлена вдоль радиуса – вектора точки и обратно пропорционально квадрату расстояния, то последнее выражение показывает, что векторное поле $\boldsymbol{F}$ является полем силы притяжения. Кроме того

$\mathbf{Curl}[\boldsymbol{F}, \{\boldsymbol{r}, \boldsymbol{\varphi}, \boldsymbol{\theta}\}, \text{"}\mathbf{Spherical}\text{"}]$

`{0,0,0}`

Легко видеть, что любое векторное поле, задаваемое в сферической системе координат в виде $(f(r),0,0)$, является безвихревым.

```
Curl[{f[r],0,0},{r,θ,φ},"Spherical"]
{0,0,0}
```

Иногда функция `Curl` возвращает разреженную матрицу, которая выглядит не совсем обычно

```
s = Curl[x + y + z,{x,y,z}]
StructuredArray[SymmetrizedArray,{3,3},-Structured Data-]
```

Чтобы преобразовать результат в обычную матрицу к нему нужно применить функцию `Normal`.

```
Normal[s]
{{0,1,-1},{-1,0,1},{1,-1,0}}
```

Между дифференциальными операторами векторного анализа имеются связи. Например, $grad\,(u\cdot v)=u\cdot grad\,(v)+v\cdot grad\,(u)$. Это тождество легко проверить. Имеем

```
Remove[u,v]
Grad[u[x,y,z] v[x,y,z],{x,y,z}] == u[x,y,z] Grad[v[x,y,z],{x,y,z}] +
                                   v[x,y,z] Grad[u[x,y,z],{x,y,z}]
True
```

Для скалярного $h(x,y,z)$ и векторного поля $\mathbf{A}(x,y,z)$ выполняются тождества $div(h\cdot\mathbf{A})=h\,div\,\mathbf{A}+\langle\mathbf{A},grad\,h\rangle$ и $rot(h\cdot\mathbf{A})=h\,rot\,\mathbf{A}-[\mathbf{A},grad\,h]$, где угловые и квадратные скобки обозначают скалярное и векторное произведение. Проверим их. Для первого тождества имеем

```
Clear[u,v,w,h];
A = {u[x,y,z],v[x,y,z],w[x,y,z]};
Div[h[x,y,z]A,{x,y,z}] == h[x,y,z]Div[A,{x,y,z}] +
                          A.Grad[h[x,y,z],{x,y,z}]//Simplify
True
```

Проверим второе тождество.

```
Curl[h[x,y,z] A,{x,y,z}] == h[x,y,z] Curl[A,{x,y,z}] -
                            A × Grad[h[x,y,z],{x,y,z}]//Simplify
True
```

Напомним, что векторное произведение (функция `Cross`) в виде × (крест) вводится как `Esc – cross – Esc`.

Имеются и другие тождества, связывающие дифференциальные операции первого порядка. Вы можете проверить их самостоятельно.

Дифференциальные операции 2 – го порядка. Вычисляя градиент скалярного поля, мы производим над ним дифференциальную операцию первого порядка. Аналогично можно говорить о дифференциальных операциях первого порядка над векторным полем (операции взятия дивергенции и ротора). Каждая операция первого порядка приводит к новому полю – векторному или скалярному. Если над ними произвести снова операцию первого порядка, то над исходным полем будет произведена операция второго порядка.

Пусть исходное поле является скалярным (функция u). Над ним можно произвести только одну дифференциальную операцию $grad\ u$. Новое поле является векторным и над ним можно произвести одну из двух операций: взять дивергенцию или ротор. Таким образом, над скалярным полем можно произвести две операции второго порядка: $rot\,grad u$ и $div\,grad\,u$. Для первой операции имеем

```
Curl[Grad[u[x, y, z], {x, y, z}], {x, y, z}]
{0, 0, 0}
```

Векторное поле, ротор которого равен нулю, называется безвихревым. Сейчас мы показали что векторное поле, которое образует градиент произвольной скалярной функции, является безвихревым.

Для второй операции в декартовой системе координат имеем

```
Div[Grad[u[x, y, z], {x, y, z}], {x, y, z}]
u^(0,0,2)[x, y, z] + u^(0,2,0)[x, y, z] + u^(2,0,0)[x, y, z]
```

Операция 2 – го порядка $div\,grad\,u$ приводит к новому скалярному полю. Она называется «оператором Лапласа» или «лапласианом» и обозначается коротко Δu. Для вычисления этой операции в *Mathematica* имеется специальная функция `Laplacian`.

```
Laplacian[u[x, y, z], {x, y, z}]
u^(0,0,2)[x, y, z] + u^(0,2,0)[x, y, z] + u^(2,0,0)[x, y, z]
Laplacian[Sqrt[x^2 + y^2 + z^2], {x, y, z}]//Simplify
       2
-----------------
Sqrt[x^2 + y^2 + z^2]
Laplacian[f[x, y], {x, y}]
f^(0,2)[x, y] + f^(2,0)[x, y]
```

Скалярное поле, лапласиан которого в некоторой области тождественно равен нулю, называется гармоническим (гармонической функцией).

```
u[x, y, z] = 1/Sqrt[x^2 + y^2 + z^2];
Laplacian[u[x, y, z], {x, y, z}]//Simplify
0
```

Т.е. функция $u=\frac{1}{\sqrt{x^2+y^2+z^2}}$ является гармонической везде, кроме точки (0,0,0), в которой она не определена.

Первым аргументом функции `Laplacian`. может быть список функций или массив большей размерности. В декартовой системе лапласиан от списка функций равен списку лапласианов этих функций.

```
Laplacian[{x^2 + y^2, Sqrt[x^2 + y^2], 1/Sqrt[x^2 + y^2], Log[x^2 + y^2]}, {x, y}]//Simplify
{4, 1/Sqrt[x^2 + y^2], 1/(x^2 + y^2)^(3/2), 0}
```

Однако в криволинейных системах координат это не так. Лапласиан даже постоянного вектора может иметь ненулевые координаты

```
Laplacian[{1, 1}, {r, φ}, "Polar"]//Simplify
```

$$\left\{-\frac{1}{r^2}, -\frac{1}{r^2}\right\}$$

Обратите внимание, что в двумерном пространстве функция $1/\sqrt{x^2+y^2}$ не является гармонической. Гармонической является функция $\ln\left(x^2+y^2\right)$.

Рассмотрим теперь дифференциальные операции 2 – го порядка над векторным полем. Операций первого порядка две: *div* **A** и *rot* **A**. Над скалярным полем *div* **A** можно произвести только операцию взятия градиента.

```
Clear[u, v, w];
A = {u[x, y, z], v[x, y, z], w[x, y, z]};
gd = Grad[Div[A, {x, y, z}], {x, y, z}]
```

$$\{w^{(1,0,1)}[x,y,z] + v^{(1,1,0)}[x,y,z] + u^{(2,0,0)}[x,y,z],$$
$$w^{(0,1,1)}[x,y,z] + v^{(0,2,0)}[x,y,z] + u^{(1,1,0)}[x,y,z],$$
$$w^{(0,0,2)}[x,y,z] + v^{(0,1,1)}[x,y,z] + u^{(1,0,1)}[x,y,z]\}$$

Над векторным полем *rot* **A** можно произвести операции *div* и *rot*; это приводит нас еще к двум операциям второго порядка: *div rot* **A** и *rot rot* **A**. Но первая операция приводит к тождественному нулю

```
Div[Curl[A, {x, y, z}], {x, y, z}]
```

```
0
```

Вторая операция *rot rot* **A** дает

```
cc = Curl[Curl[A, {x, y, z}], {x, y, z}]
```

$$\{-u^{(0,0,2)}[x,y,z] - u^{(0,2,0)}[x,y,z] + w^{(1,0,1)}[x,y,z] + v^{(1,1,0)}[x,y,z],$$
$$-v^{(0,0,2)}[x,y,z] + w^{(0,1,1)}[x,y,z] + u^{(1,1,0)}[x,y,z] - v^{(2,0,0)}[x,y,z],$$
$$v^{(0,1,1)}[x,y,z] - w^{(0,2,0)}[x,y,z] + u^{(1,0,1)}[x,y,z] - w^{(2,0,0)}[x,y,z]\}$$

Можно заметить, что между операциями *rot rot* **A** и *grad div* **A** существует связь: *grad div* $\mathbf{A} = \Delta\mathbf{A} + $ *rot rot* **A** (в декартовой системе координат). Действительно

```
gd == Laplacian[A, {x, y, z}] + cc
```

```
True
```

Мы уже говорили, что функция `Curl` умеет работать со скалярными функциями и результатом является вектор.

```
Curl[u[x, y], {x, y}]
```

$$\{-u^{(0,1)}[x,y], u^{(1,0)}[x,y]\}$$

Ее повторное использование возвращает Лапласиан, т.е. двойной ротор скалярного поля является Лапласианом этого поля. Действительно,

```
Curl[Curl[f[x, y], {x, y}], {x, y}]
```

$$f^{(0,2)}[x,y] + f^{(2,0)}[x,y]$$

Аналогичный результат имеет место для пространственного скалярного поля.

```
Curl[Curl[f[x, y, z], {x, y, z}], {x, y, z}]
```

$$f^{(0,0,2)}[x,y,z] + f^{(0,2,0)}[x,y,z] + f^{(2,0,0)}[x,y,z]$$

Выше мы показали, что дивергенция ротора векторного поля тождественно равна нулю. Но определение функции `Curl` таково, что и для скалярного поля дивергенция ротора равна нулю.

```
Div[Curl[f[x, y, z], {x, y, z}], {x, y, z}]//Normal
{0, 0, 0}
Div[Curl[f[x, y], {x, y}], {x, y}]
0
```

С другими нестандартными возможностями функций `Grad`, `Div`, `Curl` вы можете познакомиться по справочной системе. В частности, в нашем пособии мы не рассмотрели примеры того, как работают эти функции с тензорными полями.

Особую роль в приложениях играют **потенциальные поля**. Векторное поле $\vec{\mathbf{A}} = \left(A_x, A_y, A_z\right)$ называется потенциальным, если существует скалярная функция *u*, для которой $\vec{\mathbf{A}}$ служит градиентом $\vec{\mathbf{A}} = grad\, u$. Сама функция *u* называется потенциальной функцией (или потенциалом) поля $\vec{\mathbf{A}}$. Для того, чтобы поле $\vec{\mathbf{A}}$ было потенциальным необходимо и достаточно, чтобы во всей рассматриваемой области $rot\, \vec{\mathbf{A}}$ обращался в нуль.

Векторное поле, ротор которого тождественно равен нулю, называется безвихревым. Таким образом, понятия потенциального и безвихревого поля совпадают.

В курсах математического анализа доказывается, что для такого поля циркуляция по простому замкнутому контуру всегда будет нулем, а линейный интеграл по кривой, соединяющей любые две точки поля, оказывается не зависящим от формы кривой. Сама потенциальная функция *u*, с точностью до произвольного постоянного слагаемого, определяется криволинейным интегралом 2 – го рода

$$u = \int_C \left\langle \vec{\mathbf{A}}, d\,\mathbf{s} \right\rangle = \int_C \vec{\mathbf{A}} \cdot d\,\mathbf{s} = \int_C A_x\, d\,x + A_y\, d\,y + A_z\, d\,z\,,$$

взятым от некоторой фиксированной точки Q до переменной точки P рассматриваемой области по любой соединяющей эти точки кривой C. Т.о., если точки P и Q являются концевыми точками кривой C, то для потенциального поля имеет место

$$\int_C \vec{\mathbf{A}} \cdot d\,\mathbf{s} = u(P) - u(Q)$$

Потенциальную функцию *u* векторного поля можно находить по – другому. Например, пусть имеется плоское векторное поле $\vec{\mathbf{A}} = \left(A_x, A_y\right)$. Если оно потенциальное, то $\vec{\mathbf{A}} = grad\, u = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right) = \left(A_x, A_y\right)$. Т.е. $\frac{\partial u}{\partial x} = A_x$ и $\frac{\partial u}{\partial y} = A_y$.

Решив эту систему относительно *u*(*x*,*y*), вы найдете потенциальную функцию.

Пример. Показать, что векторное поле $\mathbf{F}=\left(3x^2-2xy+2, 6y^2-x^2+3\right)$ является безвихревым и найти его потенциал.

```
Clear[F1, F2, F, x, y]
F1[x_, y_] = 3x^2 - 2xy + 2;
F2[x_, y_] = 6y^2 - x^2 + 3;
F[x_, y_] = {F1[x, y], F2[x, y]};
Curl[F[x, y], {x, y}]
```

0

Поскольку $rot\,\mathbf{F}=0$, то поле безвихревое и, следовательно, имеет потенциал. Для него имеем $grad\,u=\left(\dfrac{\partial u}{\partial x}, \dfrac{\partial u}{\partial y}\right)=(F_1, F_2)=\left(3x^2-2xy+2, 6y^2-x^2+3\right)$. Тогда

```
u = ∫ F1[x, y] dx + h[y]
```

$2x + x^3 - x^2y + h[y]$

Добавление $h(y)$ здесь необходимо, поскольку постоянная интегрирования может зависеть от y. Далее

```
s = Solve[D[u, y] == F2[x, y], h'[y]]
```

$\{\{h'[y] \to 3(1 + 2y^2)\}\}$

```
h[y_] = Integrate[h'[y] /. s[[1]], y]
```

$3y + 2y^3$

Таким образом, потенциал поля равен (печатаем $\boldsymbol{u}$, а подстановка значения $h(y)$ выполняется автоматически)

```
u
```

$2x + x^3 + 3y - x^2y + 2y^3$

□

Пример. Показать, что векторное поле $\mathbf{F}=\left(3x^2+2xy, x^2+2y+z, y+3z^2\right)$ является безвихревым и определить его потенциал.

```
Remove[x, y, z, F1, F2, F3, F, h, g]
F1[x_, y_, z_] = 3x^2 + 2xy;
F2[x_, y_, z_] = x^2 + 2y + z;
F3[x_, y_, z_] = y + 3z^2;
F = {F1[x, y, z], F2[x, y, z], F3[x, y, z]}
Curl[F, {x, y, z}]
```

```
{0, 0, 0}
```

Поскольку $rot\,\mathbf{F}=0$, то поле безвихревое и, следовательно, имеет потенциал.

```
u = ∫ F1[x, y, z] dx + h[y, z]
```

$x^3 + x^2y + h[y, z]$

Добавление $h(y,z)$здесь необходимо, поскольку постоянная интегрирования может зависеть от y и z. Далее

```
s1 = Solve[D[u, y] == F2[x, y, z], ∂_y h[y, z]]
```

$\{\{h^{(1,0)}[y, z] \to 2y + z\}\}$

```
h[y_, z_] = ∫ s1[[1, 1, 2]] dy + g[z]
```

$y^2 + yz + g[z]$

Проверим, чему сейчас равно u.

$\boldsymbol{u}$

$x^3 + x^2 y + y^2 + yz + g[z]$

$\mathbf{s2 = Solve[}\boldsymbol{D[u, z]}\mathbf{ == F3[}\boldsymbol{x, y, z}\mathbf{], \partial_z}\boldsymbol{g[z]}\mathbf{]}$

$\{\{g'[z] \to 3z^2\}\}$

$\boldsymbol{g}\mathbf{[z\_] = \int s2[[1, 1, 2]]}\, \mathbb{d}\boldsymbol{z}$

z^3

Теперь

$\boldsymbol{u}$

$x^3 + x^2 y + y^2 + yz + z^3$

Итак, потенциалом поля $\boldsymbol{F}$ является функция $u(x, y, z) = x^3 + x^2 y + y^2 + y\,z + z^3$ и любая функция, отличная от u на константу.

Вычислим потенциал, используй криволинейный интеграл 2 – го рода. Потенциал u в точке (X, Y, Z) вычисляется по формуле

$$u(X, Y, Z) = \int_{(x_0, y_0, z_0)}^{(X, Y, Z)} \left(3x^2 + 2\,x\,y\right) d\,x + \left(x^2 + 2\,y + z\right) d\,y + \left(y + 3z^2\right) d\,z,$$

где в качестве начальной точки можно взять начало координат, а в качестве пути (интеграл не зависит от пути интегрирования) – любую линию, соединяющую точки $(0, 0, 0)$ и (X, Y, Z), например ломаную, изображенную на следующем рисунке.

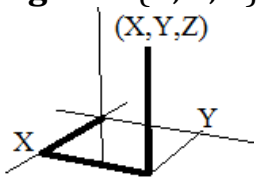
$\boldsymbol{X = 1; Y = 1; Z = 1;}$

$\mathbf{ParametricPlot3D[\{\{}\boldsymbol{tX}\mathbf{, 0, 0\}, \{}\boldsymbol{X, tY}\mathbf{, 0\}, \{}\boldsymbol{X, Y, tZ}\mathbf{\}\}, \{}\boldsymbol{t}\mathbf{, 0, 1\},}$

$\mathbf{PlotRange \to \{\{-0.5, 1.5\}, \{-0.5, 1.5\}, \{-0.5, 1.5\}\},}$

$\mathbf{PlotStyle \to \{\{Black, Thickness[0.02]\}\},}$

$\mathbf{Boxed \to False, AxesOrigin \to \{0, 0, 0\}]}$



Интеграл по такой ломаной равен сумме интегралов по отрезкам. Имеем

$\mathbf{i1 = Integrate[F1[}\boldsymbol{x, y, z}\mathbf{], \{}\boldsymbol{x}\mathbf{, 0, }\boldsymbol{X}\mathbf{\}]/.\{}\boldsymbol{y} \to \mathbf{0, }\boldsymbol{z} \to \mathbf{0\}}$

$\mathbf{i2 = Integrate[F2[}\boldsymbol{x, y, z}\mathbf{], \{}\boldsymbol{y}\mathbf{, 0, }\boldsymbol{Y}\mathbf{\}]/.\{}\boldsymbol{x} \to \boldsymbol{X}\mathbf{, }\boldsymbol{z} \to \mathbf{0\}}$

$\mathbf{i3 = Integrate[F3[}\boldsymbol{x, y, z}\mathbf{], \{}\boldsymbol{z}\mathbf{, 0, }\boldsymbol{Z}\mathbf{\}]/.\{}\boldsymbol{x} \to \boldsymbol{X}\mathbf{, }\boldsymbol{y} \to \boldsymbol{Y}\mathbf{\}}$

$\mathbf{i1 + i2 + i3}$

$X^3 + X^2 Y + Y^2 + YZ + Z^3$

Мы пришли к тому же результату, что и выше.

□

Векторное поле $\vec{\mathbf{A}}$ называется **соленоидальным**, или трубчатым, если существует векторное поле $\vec{\mathbf{B}}$ для которого $\vec{\mathbf{A}}$ служит вихрем, т.е. $\vec{\mathbf{A}} = rot\,\vec{\mathbf{B}}$. Сам вектор $\vec{\mathbf{B}}$ называют векторным потенциалом поля $\vec{\mathbf{A}}$. Для того чтобы поле

$\vec{\mathbf{A}}$ было соленоидальным, необходимо и достаточно, чтобы во всей рассматриваемой области выполнялось равенство $div\,\vec{\mathbf{A}} = 0$. Например,

```
A[x_, y_, z_] = {x y^2, x^2 y, -(x^2 + y^2) z};
Div[A[x, y, z], {x, y, z}]
0
```

и векторное поле $\boldsymbol{F}$ является соленоидальным.

Проверим, что следующее плоское векторное поле является соленоидальным.

```
A[x_, y_] = {x^2 y + y^3, x^3 - x y^2};
Div[A[x, y], {x, y}]
0
```

3.4.3 Криволинейные и поверхностные интегралы

Криволинейный интеграл 1 – го рода. Пусть в пространстве задано скалярное поле $u(x,y,z)$ и непрерывная кусочно – гладкая кривая C: $\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k}$. Криволинейным интегралом первого рода от функции $u(x,y,z)$ по кривой C называется число, равное

$$\int_C u(x,y,z)\,dl = \int_0^T u(x(t),y(t),z(t))\sqrt{x'(t)^2 + y(t)^2 + z(t)^2}\,dt \qquad (1)$$

или в векторной записи

$$\int_C u(x,y,z)\,ds = \int_0^T u(\mathbf{r}(t))\,\|\,\mathbf{r}'(t)\,\|\,dt,$$

где $\|\ \|$ обозначает норму/длину вектора.

Левая часть (1) есть обозначение интеграла первого рода, а правая есть способ его вычисления – это обычный определенный интеграл по t на [0,T]. Изначальное определение интеграла первого рода дается в терминах предела интегральных сумм. Интеграл на плоскости определяется аналогично.

Пример. Пусть вдоль винтовой линии $\mathbf{r}(t) = a\,(\mathbf{i}\cos t + \mathbf{j}\sin t) + bt\,\mathbf{k}$ распределена масса с линейной плотностью $u(x,y,z) = z^2$. Найти массу участка винтовой линии $0 \le t \le T$.

Масса кривой определяется криволинейным интегралом первого рода (1).

```
Clear[u, r, p]
r[t_] = {a Cos[t], a Sin[t], b t};
sn = Simplify[Norm[r'[t]] /. Abs[x_]^2 → x^2]
p = {x, y, z};
u[x_, y_, z_] = z^2;
M = Integrate[(u@@p /. Thread[Rule[p, r[t]]]) * sn, {t, 0, T}]
1/3 b^2 Sqrt[a^2 + b^2] T^3      (*ответ – масса кривой *)
```

Поясним некоторые элементы приведенного кода. Команда `Apply` (инфиксная форма @@) заменяет заголовок переменной $\boldsymbol{p}$ (это `List`) на u.

$\boldsymbol{u@@p}$

$u[x, y, z]$

Команда `Thread[f[args]]` «проносит» функцию `f` сквозь любые списки, которые появляется в ее аргументах `args`. Например,

$\mathbf{Thread}[\boldsymbol{f}[\{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}, \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}]]$

$\{f[a, x], f[b, y], f[c, z]\}$

В нашем примере функция `Thread` строит список подстановок

$\mathbf{Thread}[\mathbf{Rule}[\boldsymbol{p}, \boldsymbol{r}[\boldsymbol{t}]]]$

$\{x \to a\text{Cos}[t], y \to a\text{Sin}[t], z \to bt\}$

Это можно было записать еще так $\mathbf{Thread}[\boldsymbol{p} -> r[t]]$. В результате команда

$\boldsymbol{u@@p}/.\mathbf{Thread}[\mathbf{Rule}[\boldsymbol{p}, \boldsymbol{r}[\boldsymbol{t}]]]$

$u[a\text{Cos}[t], a\text{Sin}[t], bt]$

выполняет подстановку параметрических уравнений $\boldsymbol{r}[t]$ кривой в аргументы функции $u[x,y,z]$, которая требуется в интеграле (1).

Функция `Norm[r'[t]]` вычисляет норму производной радиуса – вектора $\boldsymbol{r}[t]$, и запоминает ее в переменной **sn**. Подстановка `Abs[x_]^2 → x^2` любые выражения вида $Abs[expr]^2$ заменяет выражением $expr^2$ при любом $expr$.

□

Пример. Вычислить интеграл первого рода по эллипсу $x = a\cos t,\ y = b\sin t$ $(0 \le t \le 2\pi)$ от функции $u(x, y) = \sqrt{\left(\frac{b\,x}{a}\right)^2 + \left(\frac{a\,y}{b}\right)^2}$. Имеем

$\mathbf{Clear}[\boldsymbol{u}, \boldsymbol{r}, \boldsymbol{p}]$

$\boldsymbol{r}[\mathbf{t\_}] = \{\boldsymbol{a}\mathbf{Cos}[\boldsymbol{t}], \boldsymbol{b}\mathbf{Sin}[\boldsymbol{t}]\};$

$\mathbf{sn} = \mathbf{Simplify}[\mathbf{Norm}[\boldsymbol{r}'[\boldsymbol{t}]]/.\mathbf{Abs}[\mathbf{x\_}]\mathbf{\hat{}2} \to \boldsymbol{x}\mathbf{\hat{}2}]$

$\boldsymbol{p} = \{\boldsymbol{x}, \boldsymbol{y}\};$

$\boldsymbol{u}[\boldsymbol{x\_}, \boldsymbol{y\_}] = \sqrt{\left(\frac{\boldsymbol{b}\,\boldsymbol{x}}{\boldsymbol{a}}\right)^2 + \left(\frac{\boldsymbol{a}\,\boldsymbol{y}}{\boldsymbol{b}}\right)^2};$

$\boldsymbol{L} = \mathbf{Integrate}[(\boldsymbol{u}@@\boldsymbol{p}/.\mathbf{Thread}[\mathbf{Rule}[\boldsymbol{p}, \boldsymbol{r}[\boldsymbol{t}]]]) * \mathbf{sn}, \{\boldsymbol{t}, \mathbf{0}, \mathbf{2}\boldsymbol{\pi}\}]$

$(a^2 + b^2)\pi$

□

Криволинейный интеграл 2 – го рода. Пусть в пространстве задано векторное поле $\vec{\mathbf{F}}(x, y, z) = P(x, y, z)\mathbf{i} + Q(x, y, z)\mathbf{j} + R(x, y, z)\mathbf{k}$, где P, Q, R – скалярные функции, и непрерывная кусочно – гладкая кривая C: $\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k}$. Криволинейным интегралом второго рода от вектор – функции $\vec{\mathbf{F}}(x, y, z)$ по кривой C называется число, равное

$$\int_C \langle \mathbf{F}(x, y, z), d\,\boldsymbol{l} \rangle = \int_C \mathbf{F} \cdot d\,\boldsymbol{l} = \int_C P\,dx + Q\,dy + R\,dz =$$

$$= \int_0^T \left[P(x(t), y(t), z(t))x'(t) + Q(x(t), y(t), z(t))y'(t) + R(x(t), y(t), z(t))z'(t) \right] dt \qquad (2)$$

или в векторной записи

$$\int_C \mathbf{F}\cdot d\,\boldsymbol{l} = \int_C \mathbf{F}\big(x(t), y(t), z(t)\big)\cdot \mathbf{r}'(t)\,d\,t \qquad (3)$$

Верхние записи в (2) представляют различные варианты обозначения интеграла второго рода, а нижняя представляет способ его вычисления – определенный интеграл по t в пределах [0,T]. Изначальное определение интеграла второго рода дается в терминах предела интегральных сумм. Интеграл для плоских кривых определяется аналогично.

Пример. Вычислить интеграл второго рода $I = \int_{AB} x^2 d\,x + x\,y\,d\,y$ вдоль прямолинейного отрезка , идущего из точки A(0, 0) в точку B(1, 1), и по дуге параболы $y = x^2$, соединяющей те же точки.

В первом случае $y=x$ имеем

```
Clear[F, F1, F2, p, x, y];
r1[x_] = {x, x};
p = {x, y};
F = {#1^2&, 1 * #2&};
tf = Table[F[[i]]@@p, {i, 2}]/. Thread[p → r1[x]]
Integrate[tf. r1'[x], {x, 0, 1}]
```

$$\frac{2}{3}$$

Во втором случае $y = x^2$.

```
r2[x_] = {x, x^2};
tf = Table[F[[i]]@@p, {i, 2}]/. Thread[p → r2[x]]
Integrate[tf. r2'[x], {x, 0, 1}]
```

$$\frac{11}{15}$$

Пример. Найти работу силового поля $\vec{\mathbf{F}} = x\,\mathbf{i} + y\,\mathbf{j} + z\,\mathbf{k}$ при перемещении материальной точки вдоль первого витка конической винтовой линии $x = a\,e^t \cos t$, $y = a\,e^t \sin t$, $z = a\,e^t$ из точки A(0,0,0) в точку B(a,0,a).

Заметим, что точке A соответствует значение параметра $t = -\infty$, а точке B соответствует $t = 0$. Имеем

```
Clear[F, r, p];
r[t_] = {aExp[t]Cos[t], aExp[t]Sin[t], aExp[t]};
p = {x, y, z};
F = {#1&, #2&, #3&};
tf = Table[F[[i]]@@p, {i, 3}]/. Thread[p → r[t]]
g[t_] = tf. r'[t]//Simplify
A = Integrate[g[t], {t, -∞, 0}]
```

$\{a e^t \mathrm{Cos}[t], a e^t \mathrm{Sin}[t], a e^t\}$

$2a^2 e^{2t}$

a^2

Для наглядности мы привели результаты работы последних трех строк кода. Окончательно, работа $A = a^2$. □

Криволинейный интеграл второго рода от вектора ***F***, взятый по замкнутому контуру *C*, называется циркуляцией вектора поля по данному контуру и обозначается символом $\oint_C \vec{\mathbf{F}}\, d\,\boldsymbol{l}$. Направление обхода контура указывается заранее, причем положительным считается обход против часовой стрелки.

Для плоских векторных полей $\vec{\mathbf{F}} = (P(x,y), Q(x,y))$ выполняется формула Грина

$$\oint_C P\,dx + Q\,dy = \iint_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) dx\,dy, \qquad (4)$$

где D представляет область, ограниченную кривой C. Формула (4) справедлива только в случае, когда функции *P* и *Q* непрерывны вместе со своими производными $\frac{\partial Q}{\partial x}, \frac{\partial P}{\partial y}$ в замкнутой области $\overline{D} = D \bigcup C$.

Пример. С помощью формулы Грина вычислить криволинейный интеграл $\oint_C (x+y)dx - (x-y)dy$, где C – окружность $x^2 + y^2 = R^2$. Имеем

```
Clear[F, x, y]
F[x_, y_] = {x + y, -(x - y)};
Integrate[Curl[F[x, y], {x, y}] * Boole[x^2 + y^2 <= R^2],
          {x, -R, R}, {y, -R, R}, Assumptions :> R > 0]
```

$-2\pi R^2$

□

Пример. Используя формулу Грина (4), вычислить криволинейный интеграл $\oint_C e^{2x+y}\,dx + e^{-y}\,dy$ по контуру квадрата C с вершинами (0,0), (1,0), (1,1), (0,1).

Поскольку функции *P* и *Q* непрерывны вместе со своими производными $\frac{\partial Q}{\partial x}, \frac{\partial P}{\partial y}$ в области квадрата, то формула Грина применима. Имеем

```
F[x_, y_] = {Exp[2x + y], Exp[-y]};
Integrate[Curl[F[x, y], {x, y}], {x, 0, 1}, {y, 0, 1}]
```

$-\frac{1}{2}(-1+e)^2(1+e)$

```
N[%]
-5.4890995
```

□

Поверхностный интеграл 1 – го рода. Пусть в пространстве задано скалярное поле *u*(*x*,*y*,*z*) и гладкая поверхность S: $\mathbf{r}(u,v) = x(u,v)\mathbf{i} + y(u,v)\mathbf{j} + z(u,v)\mathbf{k}$, $|\mathbf{r}'_u \times \mathbf{r}'_v| > 0$, $(u,v) \in \Omega$, где Ω – ограниченная область с кусочно – гладкой границей и $x(u,v), y(u,v), z(u,v)$ – непрерывно дифференцируемые на $\overline{\Omega}$ функции. Поверхностным интегралом первого рода называется число, равное

$$\iint_S u(x,y,z)\,dS = \iint_\Omega u(x(u,v), y(u,v), z(u,v))\,|\mathbf{r}'_u \times \mathbf{r}'_v|\,du\,dv, \qquad (5)$$

где справа стоит двойной интеграл по $(u,v)\in\Omega$. Левая часть (5) есть обозначение поверхностного интеграла первого рода, а правая дает способ его вычисления. Конечно, изначальное определение интеграла первого рода дается в терминах предела интегральных сумм.

В частности из (5) имеем следующую формулу для вычисления площади поверхности, заданной в параметрическом виде

$$S=\iint\limits_S dS=\iint\limits_\Omega \left|\mathbf{r}'_u\times\mathbf{r}'_v\right| du\,dv \qquad (6)$$

Пример. Используя формулу (6), вычислить площадь сферы радиуса R. Для этого уравнение сферы запишем в параметрическом виде $x=R\cos u\sin v$, $y=R\sin u\sin v$, $z=R\cos v$, где $0\le u\le 2\pi$ и $0\le v\le\pi$. Имеем

```
Clear[R, u, v, r1, r2]
r[u_, v_] = {R Cos[u] Sin[v], R Sin[u] Sin[v], R Cos[v]};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
nrm = Simplify[Norm[Cross[r1[u, v], r2[u, v]]] /. Abs[x_]^2 → x^2,
                                    Assumptions :→ R > 0]
S = Integrate[nrm, {u, 0, 2π}, {v, 0, π}]
R^2 √(Sin[v]^2)
4πR^2
```

Напомним, что подстановка Abs[x_]^2 → x^2 любые выражения вида $Abs[expr]^2$ заменяет выражением $expr^2$ при любом $expr$.

□

Пример. Вычислить интеграл $\oiint_S x\,y\,z\,dS$ по поверхности $\mathbf{r}(u,v)=(u\cos v,u\sin v,v)$ при $0\le u\le 1, 0\le v\le 2\pi$.

```
Clear[R, u, v, r1, r2]
r[u_, v_] = {u Cos[v], u Sin[v], v};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
n[u_, v_] = Simplify[Cross[r1[u, v], r2[u, v]]]
{Sin[v], −Cos[v], u}
nrm = Simplify[Norm[n[u, v]] /. Abs[x_]^2 -> x^2]
√(1 + u^2)
p = {x, y, z};
u[x_, y_, z_] = x y z;
S = Integrate[(u@@p /. Thread[p -> r[u, v]]) * nrm, {v, 0, 2π}, {u, 0, 1}]
1/16 π(−3√2 + ArcSinh[1])
N[%]
-0.65998325
```

□

Пример. Вычислить массу поверхности конуса $z=\sqrt{x^2+y^2}$, ограниченной сверху плоскостью $z=h$, если поверхностная плотность пропорциональна расстоянию от этой точки до начала координат, т.е. $\rho(x,y,z)=k\sqrt{x^2+y^2+z^2}$.

Масса конической поверхности может быть вычислена по формуле $m=\iint\limits_S \rho(x,y,z)dS=\iint\limits_S k\sqrt{x^2+y^2+z^2}\,dS$. Тогда имеем

```
Clear[R, x, y, r1, r2]
r[x_, y_] = {x, y, Sqrt[x^2 + y^2]};
r1[x_, y_] = D[r[x, y], x];
r2[x_, y_] = D[r[x, y], y];
n[x_, y_] = Simplify[Cross[r1[x, y], r2[x, y]]];
nrm = Simplify[Norm[n[x, y]] /. Abs[x_]^2 -> x^2]
```

$\sqrt{2}$

```
p = {x, y, z};
u[x_, y_, z_] = k Sqrt[x^2 + y^2 + z^2];
m = Integrate[(u @@ p /. Thread[p -> r[x, y]]) * nrm * Boole[x^2 + y^2 <= h^2],
    {x, -h, h}, {y, -h, h}, Assumptions :> h > 0]
```

$\frac{4}{3}h^3k\pi$

□

Поверхностный интеграл 2 – го рода. Потоком вектора $\vec{\mathbf{F}}=P\mathbf{i}+Q\mathbf{j}+R\mathbf{k}$ через ориентированную поверхность $\mathbf{S}$ называется величина, обозначаемая $\int\limits_S \langle \mathbf{F}, d\mathbf{S}\rangle$ (или $\int\limits_S \mathbf{F}\cdot d\mathbf{S}$) и определяемая при помощи равенства

$$\int\limits_S \langle \mathbf{F}, d\mathbf{S}\rangle = \int\limits_S \langle \mathbf{F}, \mathbf{n}\rangle dS = \int\limits_S \mathbf{F}\cdot\mathbf{n}\,dS, \qquad (7)$$

в правой части которого стоят поверхностные интегралы первого рода от скалярного произведения $\langle \mathbf{F}, \mathbf{n}\rangle = P\cos(\mathbf{n},\mathbf{i})+Q\cos(\mathbf{n},\mathbf{j})+R\cos(\mathbf{n},\mathbf{k})$ вектора $\vec{\mathbf{F}}$ и единичной нормали $\mathbf{n}$, определяющей ориентацию $\boldsymbol{S}$. Выражение в левой части (7) еще называют поверхностным интегралом второго рода. Изначальное определение интеграла второго рода дается в терминах предела интегральных сумм.

Если поверхность задана параметрически

$$\mathbf{r}(u,v)=x(u,v)\mathbf{i}+y(u,v)\mathbf{j}+z(u,v)\mathbf{k},\ \ |\mathbf{r}'_u\times\mathbf{r}'_v|>0,\ (u,v)\in\Omega,$$

где Ω – ограниченная область с кусочно – гладкой границей и $x(u,v), y(u,v), z(u,v)$ – непрерывно дифференцируемые на $\overline{\Omega}$ функции, то справедливо равенство

$$\int\limits_S \mathbf{F}\cdot\mathbf{n}\,dS=\int\limits_\Omega \mathbf{F}(x(u,v),y(u,v),z(u,v))\cdot\mathbf{n}(u,v)\,du\,dv, \qquad (8)$$

где в правой части стоит двойной интеграл по области $(u,v)\in\Omega$.

Пример. Вычислить интеграл $\int_S \mathbf{F}\cdot d\,\mathbf{S}$, где $\vec{\mathbf{F}} = x\,z\,\mathbf{i} + z\,\mathbf{j} + y\,x\,\mathbf{k}$ и $\mathbf{r}(u,v) = \left(u - v^2\right)\mathbf{i} + u\,v\,\mathbf{j} + \left(u^2 - v\right)\mathbf{k}$, $0 \le u \le 2$ и $1 \le v \le 3$.

```
Clear[F, r, r1, r2, n]
r[u_, v_] = {u - v^2, uv, u^2 - v};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
n[u_, v_] = Cross[r1[u, v], r2[u, v]]
p = {x, y, z};
F = {(#1#3)&, #3&, (#2#1)&};
tf = Table[F[[i]]@@p, {i, 3}]/. Thread[p → r[u, v]]
flux = Integrate[tf. n[u, v], {u, 0, 2}, {v, 1, 3}]
```

$$-\frac{6928}{15}$$

Здесь переменная **tf** содержит результат подстановки координат вектора $\boldsymbol{r}$ в выражение вектор функции, т.е. $\mathbf{F}\left(x(u,v), y(u,v), z(u,v)\right)$.

□

Пример. Найти поток вектора $\vec{\mathbf{F}} = x\,\mathbf{i} + y\,\mathbf{j} + z\,\mathbf{k}$ через часть поверхности эллипсоида $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, лежащую в первом октанте, в направлении внешней нормали.

Используем уравнение эллипсоида в параметрическом виде. Имеем.

```
Clear[F, r, r1, r2, n]
r[u_, v_] = {aCos[u]Cos[v], bCos[u]Sin[v], cSin[u]};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
n[u_, v_] = Simplify[-Cross[r1[u, v], r2[u, v]]]
```

$\{b\ c\ \mathrm{Cos}[u]^2\mathrm{Cos}[v],\ a\ c\ \mathrm{Cos}[u]^2\mathrm{Sin}[v],\ a\ b\ \mathrm{Cos}[u]\mathrm{Sin}[u]\}$

Обратите внимание, что при вычислении вектора нормали перед векторным произведением мы поставили знак минус. Это нужно, чтобы вектор **n** был внешней нормалью.

```
p = {x, y, z};
F[x_, y_, z_] = {x, y, z};
flx = Simplify[(F[x, y, z]/. Thread[p → r[u, v]]). n[u, v]];
flux = Integrate[flx, {u, 0, π/2}, {v, 0, π/2}]
```

$$\frac{1}{2}\,a\ b\ c\ \pi$$

□

Пример. Найти поток ротора векторного поля $\vec{\mathbf{F}} = \left(x^2 + y^2\right)\mathbf{i} + \left(x + z^2\right)\mathbf{j}$ через часть поверхности конуса $z^2 = x^2 + y^2$ $\left(1 \le z \le 4\right)$ в направлении внешней нормали.

Вначале составим параметрическое уравнение этой части конуса. Оно имеет вид $\mathbf{r}(u,v) = u\cos v\,\mathbf{i} + u\sin v\,\mathbf{j} + u\,\mathbf{k}$, $1 \le u \le 4$ и $0 \le v \le 2\pi$. Тогда имеем

```
Clear[F, r, r1, r2, n]
r[u_, v_] = {uCos[v], uSin[v], u};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
n[u_, v_] = Simplify[-Cross[r1[u, v], r2[u, v]]]
{uCos[v], uSin[v], -u}
p = {x, y, z};
F[x_, y_, z_] = {x^2 + y^2, x + z^2, 0};
flx = Simplify[(Curl[F[x, y, z], {x, y, z}]/. Thread[p → r[u, v]]). n[u, v]]
flux = Integrate[flx, {v, 0, 2π}, {u, 1, 4}]
-15π
```

□

Мы уже говорили, что криволинейный интеграл второго рода от вектора $\boldsymbol{F}$, взятый по замкнутому контуру C, называется циркуляцией вектора поля по данному контуру и обозначается символом $\oint_C \vec{\mathbf{F}}\, d\,\boldsymbol{l}$. Если поле плоское, то для него выполняется формула Грина. Если же поле и контур расположены в пространстве, то справедлива формула Стокса

$$\int_C \mathbf{F}\cdot d\,\boldsymbol{l} = \int_S rot\,\mathbf{F}\cdot d\,\mathbf{S}, \qquad (9)$$

где справа стоит поверхностный интеграл второго рода от ротора векторного поля $\boldsymbol{F}$, а слева – криволинейный интеграл второго рода по контуру C поверхности, ориентированному соответственно ориентации $\boldsymbol{S}$. Формула (9) выражает тот факт, что поток вектора $rot\,\vec{\mathbf{F}}$ через ориентируемую поверхность $\boldsymbol{S}$ равен циркуляции вектора $\boldsymbol{F}$ по контуру C этой поверхности.

Вспоминая формулу (8), имеем

$$\int_C \mathbf{F}\cdot d\,\boldsymbol{l} = \int_\Omega \big(rot\,\mathbf{F}\big)\big(x(u,v), y(u,v), z(u,v)\big)\cdot \mathbf{n}(u,v)\,d\,u\,d\,v \qquad (10)$$

Пример. Вычислить интеграл $\int_C \mathbf{F}\cdot d\,\boldsymbol{l}$, где $\vec{\mathbf{F}} = x\,y\,z\,\mathbf{i} + \big(z + 3x - 3y\big)\mathbf{j} + y^2\,x\mathbf{k}$ и C является окружностью радиуса R в плоскости xy. Имеем.

```
Clear[F, r, R]
r[t_] = {RCos[t], RSin[t], 0};  (* параметрическое уравнение окружности C *)
p = {x, y, z};
F[x_, y_, z_] = {x y z, z + 3x - 3y, y^2 x};
circ = Integrate[(F[x, y, z]/. Thread[p → r[t]]). r'[t], {t, 0, 2π}]
3πR^2
```

Теперь применим формулу (10). Имеем

```
Clear[F, r, r1, r2, n, R]
r[u_, v_] = {RCos[u]Sin[v], RSin[u]Sin[v], RCos[v]};
r1[u_, v_] = D[r[u, v], u];
r2[u_, v_] = D[r[u, v], v];
n[u_, v_] = Simplify[-Cross[r1[u, v], r2[u, v]]]
{R^2Cos[u]Sin[v]^2, R^2Sin[u]Sin[v]^2, R^2Cos[v]Sin[v]}
```

$\boldsymbol{p = \{x, y, z\}}$;
$\boldsymbol{F[\mathrm{x\_}, \mathrm{y\_}, \mathrm{z\_}] = \{xyz, z + 3x - 3y, y^2 x\}}$;
$\boldsymbol{\mathbf{flx} = \mathbf{Simplify}[(\mathbf{Curl}[F[x, y, z], \{x, y, z\}]/.\mathbf{Thread}[p \to r[u, v]]).n[u, v]]}$;
$\boldsymbol{\mathbf{flux} = \mathbf{Integrate}[\mathbf{flx}, \{v, 0, \pi/2\}, \{u, 0, 2\pi\}]}$
$3\pi R^2$

□

Пример. Для векторного поля $F = \left(e\, x^3 - 3\, x\, y + z^3, 2\, z^3 - x\, z^2 + y^4, 6\, y + 2\, z^3 x^2\right)$ и поверхности S, являющейся частью параболоида $z = x^2 + y^2$ $\left(z \le 9\right)$, проверить теорему Стокса.

Вначале вычислим циркуляцию как криволинейный интеграл второго рода. Кривой C, по которой вычисляется циркуляция, является окружность радиуса 3, расположенная в плоскости $z = 9$. Записываем ее параметрическое уравнение и используем формулу (3). Имеем

$\boldsymbol{\mathbf{Clear}[t, r, F]}$
$\boldsymbol{p = \{x, y, z\}}$;
$\boldsymbol{r[\mathrm{t\_}] = \{3\mathbf{Cos}[t], 3\mathbf{Sin}[t], 9\}}$;
$\boldsymbol{F[\mathrm{x\_}, \mathrm{y\_}, \mathrm{z\_}] = \{x^3 e - 3xy + z^3, 2z^3 - xz^2 + y^4, 6y + 2z^3 x^2\}}$;
$\boldsymbol{\mathbf{circ} = \mathbf{Integrate}[(F[x, y, z]/.\mathbf{Thread}[p \to r[t]]).r'[t], \{t, 0, 2\pi\}]}$
$-729\,\pi$

В формуле Стокса (10) в качестве поверхности, натянутой на кривую C, используем часть поверхности параболоида $z = x^2 + y^2$ $\left(z \le 9\right)$. Имеем

$\boldsymbol{\mathbf{Clear}[r, x, y]}$
$\boldsymbol{r[\mathrm{x\_}, \mathrm{y\_}] = \{x, y, x^2 + y^2\}}$;
$\boldsymbol{\mathbf{r1}[\mathrm{x\_}, \mathrm{y\_}] = D[r[x, y], x]}$;
$\boldsymbol{\mathbf{r2}[\mathrm{x\_}, \mathrm{y\_}] = D[r[x, y], y]}$;
$\boldsymbol{n[\mathrm{x\_}, \mathrm{y\_}] = \mathbf{Simplify}[\mathbf{Cross}[\mathbf{r1}[x, y], \mathbf{r2}[x, y]]]}$
$\{-2x, -2y, 1\}$
$\boldsymbol{\mathbf{flx} = \mathbf{Simplify}[(\mathbf{Curl}[F[x, y, z], \{x, y, z\}]/.z \to x^2 + y^2).n[x, y]]}$;
$\boldsymbol{\mathbf{flux} = \mathbf{Integrate}[\mathbf{flx} * \mathbf{Boole}[x^2 + y^2 \le 9], \{x, -3, 3\}, \{y, -3, 3\}]}$
-729π

Значения циркуляции, вычисленные разными способами, совпали.

□

Пусть V будет областью в $\mathbf{R}^3$ с границей ∂V, которая является кусочно – гладкой поверхностью, ориентированной так, что ее вектор нормали направлен наружу этой области. И пусть $\mathbf{F}(x, y, z))$ является векторным полем с непрерывными частными производными в области $\overline{V}$. Тогда выполняется формула Гаусса – Остроградского

$$\iint\limits_{\partial V} \mathbf{F} \cdot d\,\mathbf{S} = \iiint\limits_{V} div\,\mathbf{F}\, dV \qquad (11)$$

Она говорит, что поток векторного поля $\boldsymbol{F}$ через замкнутую поверхность ∂V, лежащую в этом поле, в направлении ее внешней нормали, равен тройному

(объемному) интегралу по области V, ограниченной этой поверхностью, от дивергенции этого поля.

Пример. Вычислить поток векторного поля $\vec{\mathbf{F}} = \left(x, y^2, y+z\right)$ через поверхность S тела, являющегося частью цилиндра $x^2 + y^2 \le 4$ расположенной между плоскостями $z = x$ и $z = 8$. Имеем

```
Clear[x, y, z, F]
F[x_, y_, z_] = {x, y^2, y + z};
Integrate[Div[F[x, y, z], {x, y, z}] * Boole[x^2 + y^2 <= 4 && x <= z && z <= 8],
                                {x, -2, 2}, {y, -2, 2}, {z, -∞, ∞}]
```

64π

□

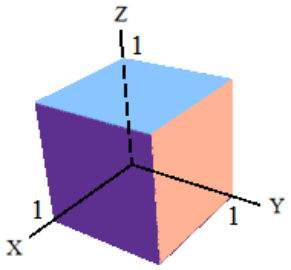
Пример. Используя теорему Гаусса – Остроградского, найти поток вектора $\mathbf{F} = x^3\,\mathbf{i} + y^3\,\mathbf{j} + R^2 z\,\mathbf{k}$ через всю поверхность тела $\frac{H}{R^2}\left(x^2+y^2\right) \le z \le H$.

```
Clear[x, y, z, F, R, H]
F[x_, y_, z_] = {x^3, y^3, R^2 z};
div = Div[F[x, y, z], {x, y, z}]
Integrate[div * Boole[(H/R^2) (x^2 + y^2) <= z <= H],
        {x, -∞, ∞}, {y, -∞, ∞}, {z, -∞, ∞}, Assumptions :→ H > 0 && R > 0]
```

$R^2 + 3x^2 + 3y^2$

$H\pi R^4$

Пример. Для векторного поля $\mathbf{F} = x\,y\,z\,\mathbf{i} + \left(x^2+y^2+z^2\right)\mathbf{j} + \left(x\,y + y\,z + x\,z\right)\mathbf{k}$ проверить формулу (11), если областью V является единичный куб, показанный на следующем рисунке.



Вначале вычисляем поток векторного поля через границу куба. Для этого вычислим поток поля через каждую его грань, учитывая направление внешней нормали. Имеем

```
Clear[x, y, z, F]
F[x_, y_, z_] = {x y z, x^2 + y^2 + z^2, x y + y z + x z};
nz = {0, 0, 1};
Iz0 = Integrate[-F[x, y, 0].nz, {x, 0, 1}, {y, 0, 1}]
Iz1 = Integrate[F[x, y, 1].nz, {x, 0, 1}, {y, 0, 1}]
```

$-\frac{1}{4}$

$\frac{5}{4}$

Здесь через **nz** мы обозначили единичный вектор, направленный вдоль оси Z, а через **Iz0** и **Iz1** потоки поля через грани куба z=0 и z=1. При этом в интеграле **Iz0** мы учли, что вектор нормали направлен в сторону, противоположную вектору **nz.** Аналогичные обозначения мы используем ниже для других граней.

$\mathbf{nx} = \{\mathbf{1}, \mathbf{0}, \mathbf{0}\};$

$\mathbf{Ix0} = \mathbf{Integrate}[-\boldsymbol{F}[\mathbf{0}, \boldsymbol{y}, \boldsymbol{z}].\,\mathbf{nx}, \{\boldsymbol{y}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{z}, \mathbf{0}, \mathbf{1}\}]$

$\mathbf{Ix1} = \mathbf{Integrate}[\boldsymbol{F}[\mathbf{1}, \boldsymbol{y}, \boldsymbol{z}].\,\mathbf{nx}, \{\boldsymbol{y}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{z}, \mathbf{0}, \mathbf{1}\}]$

$\mathbf{ny} = \{\mathbf{0}, \mathbf{1}, \mathbf{0}\};$

$\mathbf{Iy0} = \mathbf{Integrate}[-\boldsymbol{F}[\boldsymbol{x}, \mathbf{0}, \boldsymbol{z}].\,\mathbf{ny}, \{\boldsymbol{x}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{z}, \mathbf{0}, \mathbf{1}\}]$

$\mathbf{Iy1} = \mathbf{Integrate}[\boldsymbol{F}[\boldsymbol{x}, \mathbf{1}, \boldsymbol{z}].\,\mathbf{ny}, \{\boldsymbol{x}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{z}, \mathbf{0}, \mathbf{1}\}]$

$\mathbf{flux1} = \mathbf{Iz0} + \mathbf{Iz1} + \mathbf{Ix0} + \mathbf{Ix1} + \mathbf{Iy0} + \mathbf{Iy1}$

$\frac{9}{4}$

Теперь используем формулу (11). Имеем

$\mathbf{flux2} = \mathbf{Integrate}[\mathbf{Div}[\boldsymbol{F}[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}], \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}], \{\boldsymbol{x}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{y}, \mathbf{0}, \mathbf{1}\}, \{\boldsymbol{z}, \mathbf{0}, \mathbf{1}\}]$

$\frac{9}{4}$

Значения совпали, но вычисления вторым способом значительно короче.

3.4.4 Криволинейные системы координат

В системе *Mathematica* имеется несколько функций, предназначенных для преобразования математических понятий из одной системы координат (СК) в другую.

Формулы перехода из одной системы координат в другую возвращает функция `CoordinateTransform`.

Функция `CoordinateTransformData` вычисляет специфические понятия, связанные с переходом из одной СК в другую. Это формулы перехода, матрица Якоби, якобиан (определитель матрицы Якоби), обратная матрица Якоби и некоторые другие.

Функция `TransformedField` выполняет преобразование скалярных и векторных полей при переходе из одной системы координат в другую.

Функция `CoordinateChartData` возвращает величины, характеризующие систему координат, например, метрические коэффициенты.

Команда `CoordinateTransform[преобразование,точка]` возвращает координаты точки в СК, заданной в «`преобразовании`». Например,

$\{\boldsymbol{x}, \boldsymbol{y}\} = \mathbf{CoordinateTransform}[\text{"}\mathbf{Polar}\text{"} \rightarrow \text{"}\mathbf{Cartesian}\text{"}, \{\boldsymbol{r}, \boldsymbol{\varphi}\}]$

$\{r\mathrm{Cos}[\varphi], r\mathrm{Sin}[\varphi]\}$

Здесь первый аргумент `"Polar"→"Cartesian"` определяет имена СК (в нашем примере выполняется переход из полярной системы в декартову); второй аргумент `{r,φ}` задает координаты точки в исходной СК. Имена результирующих координат вы можете задавать произвольно в левой части. В нашем случае мы определили, что переменные в декартовой/результирующей системе координат будут иметь имена *x* и *y*.

Второй аргумент может иметь конкретные/числовые значения координат; также он может быть списком координат точек.

CoordinateTransform[{"**Cartesian**"−> "**Polar**"}, {{**1**, **0**}, {**1**, **1**}, {**0**, **1**}}]

$\{\{1,0\}, \{\sqrt{2}, \frac{\pi}{4}\}, \{1, \frac{\pi}{2}\}\}$

CoordinateTransform["**Spherical**" → "**Cartesian**", {**1**, $\boldsymbol{\pi}$/**4**, $\boldsymbol{\pi}$/**2**}]

$\{0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\}$

Пусть, например, кривая задана параметрическими уравнениями в полярной системе координат. Используя функцию `CoordinateTransform`, можно получить параметрические уравнения кривой в декартовых координатах, а затем построить ее график.

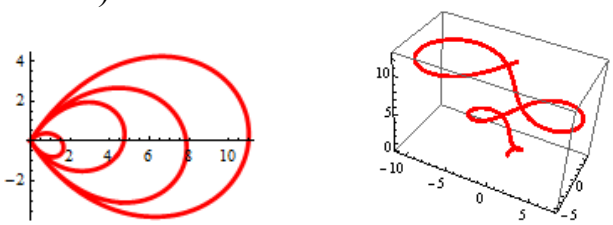
plr[**t**_] := {$\boldsymbol{t}$ **Sin**[$\boldsymbol{t}$]^**2**, **Cos**[$\boldsymbol{t}$]};

cart[**t**_] = **CoordinateTransform**["**Polar**" → "**Cartesian**", **plr**[$\boldsymbol{t}$]]

$\{t\text{Cos}[\text{Cos}[t]]\text{Sin}[t]^2, t\text{Sin}[t]^2\text{Sin}[\text{Cos}[t]]\}$

ParametricPlot[**cart**[$\boldsymbol{t}$], {$\boldsymbol{t}$, **0**, **4**$\boldsymbol{\pi}$}, **PlotStyle** → {**Red**, **Thickness**[**0.01**]}]

(следующий рисунок слева).



Вот аналогичный пример кривой, заданной параметрически в цилиндрической системе координат

cyl[**t**_] := {$\boldsymbol{t}$**Sin**[$\boldsymbol{t}$], **Cos**[$\boldsymbol{t}$], $\boldsymbol{t}$};

cart[**t**_] = **CoordinateTransform**["**Cylindrical**" → "**Cartesian**", **cyl**[$\boldsymbol{t}$]]

$\{t\text{Cos}[\text{Cos}[t]]\text{Sin}[t], t\text{Sin}[t]\text{Sin}[\text{Cos}[t]], t\}$

ParametricPlot3D[**cart**[$\boldsymbol{t}$], {$\boldsymbol{t}$, **0**, **4**$\boldsymbol{\pi}$}, **PlotStyle** → {**Red**, **Thickness**[**0.01**]}]

(предыдущий рисунок справа).

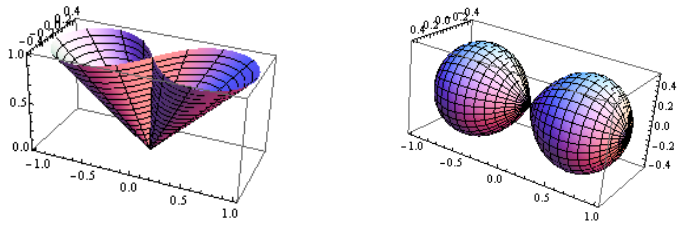
В цилиндрической (или другой допустимой) системе координат вы можете задать параметрическое уравнение поверхности, преобразовать его к декартовым координатам и построить график поверхности.

cyl[**u**_, **v**_] := {$\boldsymbol{u}$**Sin**[$\boldsymbol{v}$], **Cos**[$\boldsymbol{v}$], $\boldsymbol{u}$};

cart[**u**_, **v**_] = **CoordinateTransform**["**Cylindrical**" → "**Cartesian**", **cyl**[$\boldsymbol{u}$, $\boldsymbol{v}$]]

$\{u\text{Cos}[\text{Cos}[v]]\text{Sin}[v], u\text{Sin}[v]\text{Sin}[\text{Cos}[v]], u\}$

ParametricPlot3D[**cart**[$\boldsymbol{u}$, $\boldsymbol{v}$], {$\boldsymbol{u}$, **0**, **1**}, {$\boldsymbol{v}$, **0**, **2**$\boldsymbol{\pi}$}] (* следующий рис. слева *)



Аналогично, зададим поверхность в параметрическими уравнениями в сферической системе координат и построим ее, перейдя к параметрическим уравнениям в декартовой системе.

$$\mathbf{sph}[\mathbf{u\_}, \mathbf{v\_}] := \{\mathbf{Sin}[v], \mathbf{Cos}[v], u\};$$
$$\mathbf{cart}[\mathbf{u\_}, \mathbf{v\_}] = \mathbf{CoordinateTransform}[\text{"}\mathbf{Spherical}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \mathbf{sph}[u, v]]$$
$$\mathbf{ParametricPlot3D}[\mathbf{cart}[u, v], \{u, \mathbf{0}, \pi\}, \{v, \mathbf{0}, \mathbf{2}\pi\}, \mathbf{Mesh} \to \{\mathbf{15}, \mathbf{60}\}]$$

(предыдущий рисунок справа).

Функция `CoordinateTransformData` вычисляет значения величин, связанных с переходом из одной СК в другую. Это могут быть формулы перехода, матрица Якоби, якобиан (определитель матрицы Якоби), обратная матрица Якоби и некоторые другие.

Для получения формул перехода из одной системы координат в другую можно использовать функцию `CoordinateTransformData`

$$\{x, y\} = \mathbf{CoordinateTransformData}[\text{"}\mathbf{Polar}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \text{"}\mathbf{Mapping}\text{"}, \{r, \varphi\}]$$

$$\{r\mathrm{Cos}[\varphi], r\mathrm{Sin}[\varphi]\}$$

Здесь первый аргумент `"Polar"→"Cartesian"` определяет имена систем координат; второй аргумент определяет, что функция будет возвращать (`Mapping` говорит, что функция вернет формулы перехода); третий аргумент `{r,φ}` задает имена отображаемых переменных (в нашем случае имена переменных в полярной системе координат). Имена результирующих координат можно задавать в левой части.

$$\mathbf{CoordinateTransformData}[\text{"}\mathbf{Cartesian}\text{"} \to \text{"}\mathbf{Spherical}\text{"}, \text{"}\mathbf{Mapping}\text{"}, \{x, y, z\}]$$

$$\{\sqrt{x^2 + y^2 + z^2}, \mathrm{ArcTan}[z, \sqrt{x^2 + y^2}], \mathrm{ArcTan}[x, y]\}$$

Если третий аргумент не передавать, то будут возвращены «чистые» функции.

$$\mathbf{CoordinateTransformData}[\text{"}\mathbf{Polar}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \text{"}\mathbf{Mapping}\text{"}]$$

{Cos[#1[[2]]] #1[[1]], Sin[#1[[2]]] #1[[1]]}&

Если вы хотите получить матрицу Якоби замены переменных, то второй аргумент должен называться `"MappingJacobian"`

$$\mathbf{MJ} = \mathbf{CoordinateTransformData}[\text{"}\mathbf{Polar}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \text{"}\mathbf{MappingJacobian}\text{"}, \{r, \varphi\}];$$

$$\mathbf{MJ//MatrixForm}$$

$$\begin{pmatrix} \mathrm{Cos}[\varphi] & -r\mathrm{Sin}[\varphi] \\ \mathrm{Sin}[\varphi] & r\mathrm{Cos}[\varphi] \end{pmatrix}$$

Если вы хотите получить Якобиан (детерминант матрицы Якоби), то второй аргумент должен называться `"MappingJacobianDeterminant"`

$$\mathbf{Jac} = \mathbf{CoordinateTransformData}[\text{"}\mathbf{Polar}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \text{"}\mathbf{MappingJacobianDeterminant}\text{"}, \{r, \varphi\}]$$

r

Второй аргумент может быть `"InverseMappingJacobian"`. В этом случае вы получите обратную матрицу Якоби.

$$\mathbf{CoordinateTransformData}[\text{"}\mathbf{Polar}\text{"} \to \text{"}\mathbf{Cartesian}\text{"}, \text{"}\mathbf{InverseMappingJacobian}\text{"}, \{r, \varphi\}]\mathbf{//MatrixForm}$$

$$\begin{pmatrix} \mathrm{Cos}[\varphi] & \mathrm{Sin}[\varphi] \\ -\dfrac{\mathrm{Sin}[\varphi]}{r} & \dfrac{\mathrm{Cos}[\varphi]}{r} \end{pmatrix}$$

Имена всех свойств, которые могут стоять вторым аргументом, можно получить командой

CoordinateTransformData["**Properties**"]

```
{InverseMappingJacobian,Mapping,
 MappingJacobian,MappingJacobianDeterminant,
 OrthonormalBasisRotation,StandardName}
```

Функция `TransformedField` умеет преобразовывать скалярные и векторные поля при переходе из одной системы координат в другую.

Clear[x, y, r, φ]

TransformedField["**Polar**" → "**Cartesian**", r^2**Cos**[θ], {r, θ} → {x, y}]

$x\sqrt{x^2 + y^2}$

TransformedField["**Cartesian**" → "**Polar**", $x^2 + y^2$,
{x, y} → {r, φ}]//**Simplify**

r^2

В этих двух примерах вторым аргументом стояло выражение (скалярное поле), которое надо было преобразовать к новой системе координат.

В следующем примере преобразуется векторное поле (т.е. вектор) (x, y).

TransformedField[{"**Cartesian**" → "**Polar**", **2**}, {x, y},
{x, y} → {r, φ}]//**Simpl**ify

```
{r,0}
```

В полярной системе координат это поле определяется как $(r, 0)$.

Важно помнить, что имена переменных систем координат должны различаться. Например

Clear[x, y, z, r, φ, z];

TransformedField["**Cartesian**" → "**Cylindrical**", $x^2 + y^2 + z^2$,
{x, y, z} → {r, φ, ζ}]//**Simplify**

$r^2 + \zeta^2$

Здесь мы вынуждены вертикальную координату z цилиндрической системы назвать ζ, иначе будет сообщение об ошибке.

Вторым аргументом может стоять постоянный вектор

TransformedField["**Polar**" → "**Cartesian**", {**1, 0**}, {r, φ}−> {x, y}]

$\{\frac{x}{\sqrt{x^2 + y^2}}, \frac{y}{\sqrt{x^2 + y^2}}\}$

TransformedField["**Cartesian**"−> "**Polar**", {**1, 1**}, {x, y}−> {r, φ}]

$\{\text{Cos}[\varphi] + \text{Sin}[\varphi], \text{Cos}[\varphi] - \text{Sin}[\varphi]\}$

Пример. Потенциал поля тяготения (с точностью до постоянного множителя) задан в сферических координатах. Вычислить его градиент в декартовой СК.

u[r_, θ_, φ_] = **1**/r;

sphGrad = **Grad**[u[r, θ, φ], {r, θ, φ}, "**Spherical**"]

$\{-\frac{1}{r^2}, 0, 0\}$

TransformedField["**Spherical**" → "**Cartesian**", **sphGrad**, {***r***, ***θ***, ***φ***} → {***x***, ***y***, ***z***}]

$$\left\{-\frac{x}{(x^2+y^2+z^2)^{3/2}},-\frac{y}{(x^2+y^2+z^2)^{3/2}},-\frac{z}{(x^2+y^2+z^2)^{3/2}}\right\}$$

Можно вначале преобразовать потенциал `u` к декартовым координатам, а потом вычислить его градиент.

U[x_, y_, z_] = **TransformedField**["**Spherical**" → "**Cartesian**", ***u***[***r***, ***θ***, ***φ***], {***r***, ***θ***, ***φ***} → {***x***, ***y***, ***z***}]

$$\frac{1}{\sqrt{x^2+y^2+z^2}}$$

Grad[***U***[***x***, ***y***, ***z***], {***x***, ***y***, ***z***}, "**Cartesian**"]

$$\left\{-\frac{x}{(x^2+y^2+z^2)^{3/2}},-\frac{y}{(x^2+y^2+z^2)^{3/2}},-\frac{z}{(x^2+y^2+z^2)^{3/2}}\right\}$$

□

Пример. Потенциал поля задан в сферических координатах $u(r,\theta,\varphi)=\frac{p\cos\theta}{r^2}$ (электростатический потенциал диполя *p*, расположенного в начале координат и ориентированного вдоль оси z). Построить линии тока соответствующего векторного поля (силовые линии электрического поля) в плоскости YZ.

Векторное силовое поле вычисляется как градиент потенциала. Имеем

p = **1**; ***u***[r_, **θ**_, **φ**_] = $\frac{\boldsymbol{p}\mathbf{Cos}[\boldsymbol{\theta}]}{\boldsymbol{r}^2}$;

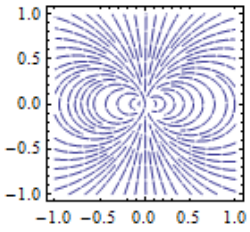
Esph = −**Grad**[***u***[***r***, ***θ***, ***φ***], {***r***, ***θ***, ***φ***}, "**Spherical**"]

$$\left\{\frac{2\mathrm{Cos}[\theta]}{r^3},\frac{\mathrm{Sin}[\theta]}{r^3},0\right\}$$

Ecart = **TransformedField**["**Spherical**" → "**Cartesian**", **Esph**, {***r***, ***θ***, ***φ***} → {***x***, ***y***, ***z***}]

$$\left\{\frac{3xz}{(x^2+y^2+z^2)^{5/2}},\frac{3yz}{(x^2+y^2+z^2)^{5/2}},-\frac{x^2+y^2}{(x^2+y^2+z^2)^{5/2}}+\frac{2z^2}{(x^2+y^2+z^2)^{5/2}}\right\}$$

StreamPlot[**Rest**[**Ecart**]/. ***x*** → **0**, {***y***, −**1**, **1**}, {***z***, −**1**, **1**}]



Вначале мы вычислили силовое поле **Esph** в сферической СК. Затем преобразовали его к декартовым координатам. После этого отбросили первую координату векторного поля **Ecart** (функция `Rest` отбрасывает первый элемент списка), а в оставшиеся координатные функции подставили *x*=0.

□

Пример. Вычислить массу поверхности конуса $z=\sqrt{x^2+y^2}$, ограниченной сверху плоскостью $z=h$, если поверхностная плотность пропорциональна расстоянию от точки поверхности до начала координат, т.е. $\rho(x,y,z)=k\sqrt{x^2+y^2+z^2}$. Мы уже решали эту задачу в предыдущем пункте, выполняя все вычисления в декартовой СК.

Масса любой поверхности может быть вычислена по формуле $m=\iint\limits_S \rho(x,y,z)\,dS$. Если поверхность S задана параметрическими уравнениями $\mathbf{r}(u,v)=x(u,v)\mathbf{i}+y(u,v)\mathbf{j}+z(u,v)\mathbf{k}$, $|\mathbf{r}'_u\times\mathbf{r}'_v|>0$, $(u,v)\in\Omega$, то поверхностный интеграл первого рода сводится к двойному интегралу по Ω. В нашем случае мы имеем

$$m=\iint\limits_{\Omega} k\sqrt{x(u,v)^2+y(u,v)^2+z(u,v)^2}\,|\mathbf{r}'_u\times\mathbf{r}'_v|\,du\,dv$$

Параметрическое уравнение конуса можем представить в виде $\mathbf{r}(x,y)=\left(x,y,\sqrt{x^2+y^2}\right)$. Вначале вычисляем подынтегральное выражение в декартовых координатах (в коде оно обозначено как $\boldsymbol{v}$).

```
Clear[R, x, y, r1, r2]
r[x_, y_] = {x, y, Sqrt[x^2 + y^2]};
r1[x_, y_] = D[r[x, y], x];
r2[x_, y_] = D[r[x, y], y];
n[x_, y_] = Simplify[Cross[r1[x, y], r2[x, y]]];
nrm = Simplify[Norm[n[x, y]] /. Abs[x_]^2 -> x^2]
p = {x, y, z};
u[x_, y_, z_] = k Sqrt[x^2 + y^2 + z^2];
v = Simplify[(u @@ p /. Thread[p -> r[x, y]]) * nrm]
```

$2k\sqrt{x^2+y^2}$

Теперь преобразуем $\boldsymbol{v}$ к полярным координатам

```
vpol = Simplify[TransformedField["Cartesian" -> "Polar", v,
                {x, y} -> {r, φ}], Assumptions :> r > 0]
```

$2k\,r$

Вычисляем якобиан замены переменных

```
jac = CoordinateTransformData["Polar" -> "Cartesian",
        "MappingJacobianDeterminant", {r, φ}]
```

r

Теперь вычисляем двойной интеграл по кругу $r\le h$ в полярных координатах

```
m = Integrate[vpol * jac, {φ, 0, 2π}, {r, 0, h}]
```

$\frac{4}{3}h^3k\pi$

□

Функция `CoordinateChartData` возвращает величины, характеризующие систему координат, например, метрические коэффициенты. В формате `CoordinateChartData["Сист. координат","свойство","точка"]` вычисляется значение «свойства» системы координат в `точке`. Например, метрические коэффициенты СК вычисляются командой

CoordinateChartData["**Cartesian**", "**Metric**", {x, y, z}]//**MatrixForm**

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

CoordinateChartData["**Polar**", "**Metric**", {r, φ}]//**MatrixForm**

$$\begin{pmatrix} 1 & 0 \\ 0 & r^2 \end{pmatrix}$$

CoordinateChartData["**Spherical**", "**Metric**", {r, θ, φ}]//**MatrixForm**

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2 \text{Sin}[\theta]^2 \end{pmatrix}$$

Список названий свойств (второй аргумент), которые можно узнать о СК, можно получить командой

CoordinateChartData["**Properties**"]

{… , Dimension, InverseMetric, Metric, … }

Вот примеры получения значений некоторых свойств СК.

CoordinateChartData["**Cylindrical**", "**Dimension**"]

3

CoordinateChartData["**Polar**", "**ScaleFactors**", {r, θ}]

$\{1, r\}$

CoordinateChartData["**Polar**", "**CoordinateRangeAssumptions**", {r, φ}]

$r > 0 \;\&\&\; -\pi < \varphi \leq \pi$

Если третий аргумент не задан, то обычно возвращается «чистая» функция.

test = CoordinateChartData["**Polar**", "**CoordinateRangeAssumptions**"]

#1[[1]]>0 && –π < #1[[2]] ≤ π &

Последнее свойство можно использовать для проверки того, правильно ли заданы координаты точки в соответствующей СК.

test[{**3**, $\boldsymbol{\pi/2}$}]

`True`

test[{**3**, $\boldsymbol{3\pi/2}$}]

`False`

test = CoordinateChartData[{"**Spherical**", **3**},
"**CoordinateRangeAssumptions**"]

#1[[1]]>0 && 0 < #1[[2]] < π && – π < #1[[3]] ≤ π &

test[{**3**, $\boldsymbol{3\pi/2}$, **0**}]

`False`

Вычислим коэффициент пропорциональности между квадратом дифференциала $d\,s$ длины дуги кривой и квадратом дифференциала $d\,t$ параметра кривой ($d\,s^2 = \left|\mathbf{r}'(t)\right|^2 d\,t^2$) в декартовой СК.

```
r[t_] = {x[t], y[t], z[t]};        (* параметрическое уравнение кривой *)
metric[t_] = CoordinateChartData[{"Cartesian", 3}, "Metric"][r[t]]
{{1,0,0},{0,1,0},{0,0,1}}
r'[t].metric[t].r'[t]
x'[t]^2 + y'[t]^2 + z'[t]^2
```

Вычислим аналогичный коэффициент пропорциональности для кривой $r = r(\varphi)$ в полярных координатах.

```
Clear[r]
R[φ_] = {r[φ], φ};
metric[φ_] = CoordinateChartData[{"Polar", 2}, "Metric"][R[φ]]
R'[φ].metric[φ].R'[φ]
{{1,0},{0,r[φ]^2}}
r[φ]^2 + r'[φ]^2
```

Т.е. мы получили, что $d\,s = \sqrt{r^2(\varphi) + r'^2(\varphi)}d\,\varphi$.

Параметр `"VolumeFactor"` позволяет вычислить числовой множитель для элемента объема. Например

```
V = CoordinateChartData[{"Spherical", 3}, "VolumeFactor", {r, θ, φ}]
r^2 Sin[θ]
```

Зная его, можно вычислять объемы тел, границы которых задаются в сферической СК. Например, объем шара теперь можно вычислить по формуле

```
∫_0^R ∫_0^π ∫_0^2π V dφ dθ dr
4πR^3/3
```

Проверим формулу площади круга. Имеем

```
s = CoordinateChartData[{"Polar", 2}, "VolumeFactor", {r, φ}]
r
∫_0^R ∫_0^2π s dφ dr
πR^2
```

Заметим, что иногда при указании системы координат следует задавать размерность пространства. Для этого вместо имени СК надо указать список, составленный из имени СК и размерности. Например, в последней команде полярную систему координат мы задали как {"Polar",2}.

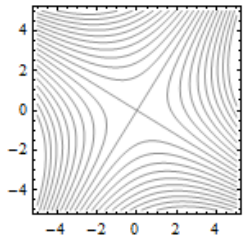
Приведенные в этом пункте функции умеют работать с большим количеством систем координат. Читатель может познакомиться с ними самостоятельно по справочной системе.

3.4.5 Графическое представление векторных полей.

Способы графического представления скалярных полей (скалярных функций двух и трех переменных) описаны нами ранее в главе «Графические возможности системы *Mathematica*». Напомним одну из таких функций.

$\boldsymbol{f[x\_, y\_] = x^2 + xy - y^2;}$

$\textbf{ContourPlot}[f[x, y], \{x, -5, 5\}, \{y, -5, 5\},$
$\textbf{ContourShading} \rightarrow \textbf{False}, \ \textbf{Contours} \rightarrow \textbf{20}]$

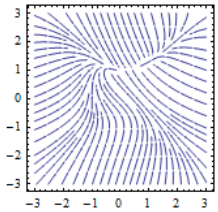


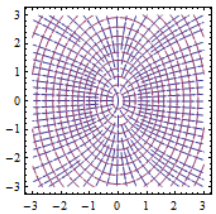
Здесь мы рассмотрим функции, которые выполняют визуализацию векторных полей (плоских и пространственных). Хотя они имеют дело с векторными данными, но по своим возможностям, форматам использования и опциям оформления они близки к `ContourPlot`.

Для графического представления плоских векторных полей можно использовать функцию `StreamPlot`. Она строит линии тока (векторные линии, силовые линии) поля $\vec{\mathbf{F}} = \left(P(x, y), Q(x, y)\right)$. Линии тока никогда не пересекаются, и через каждую точку проходит одна линия за исключением тех точек, где поле не определено или равно нулю. В декартовых координатах дифференциальные уравнения линий тока имеют вид $\dfrac{dx}{P(x, y)} = \dfrac{dy}{Q(x, y)}$.

В формате `StreamPlot[{P,Q},{x,`x_{min}`,`x_{max}`},{y`$_{min}$`,y`$_{max}$`}]` строится график линий тока векторного поля $\left(P(x, y), Q(x, y)\right)$ на плоскости xy.

$\textbf{StreamPlot}[\{-1 - x^3 + y^2, 1 + x^2 - y^3\}, \{x, -3, 3\}, \{y, -3, 3\}]$
(следующий рисунок слева)





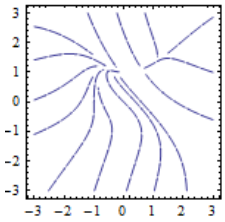
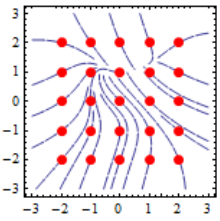
Можно строить линии тока нескольких векторных полей на одном графике.

$\textbf{StreamPlot}[\{\{x, y/2\}, \{y/2, -x\}\}, \{x, -3, 3\}, \{y, -3, 3\}]$
(предыдущий рисунок справа).

Опция `StreamPoints` позволяет указать количество линий тока, которое будет нарисовано или задать множество точек, через которые линии тока будет проходить.

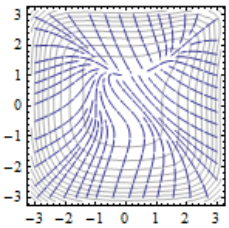
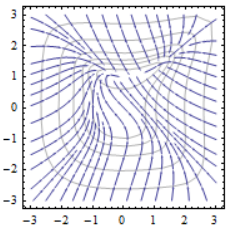
$\textbf{StreamPlot}[\{-1 - x^3 + y^2, 1 + x^2 - y^3\}, \{x, -3, 3\}, \{y, -3, 3\},$
$\textbf{StreamPoints} \rightarrow \textbf{15}]$ (*следующий рисунок слева *)
$\textbf{points} = \textbf{Flatten}[\textbf{Table}[\{x, y\}, \{x, -2, 2, 1\}, \{y, -2, 2, 1\}], 1]$

StreamPlot[{$-1-x^3+y^2, 1+x^2-y^3$}, {$x,-3,3$}, {$y,-3,3$},
Epilog → {**Red**, **PointSize**[**Large**], **Point**[**points**]},
StreamPoints → **points**] (*следующий рисунок справа *)

Опция `Mesh->n` позволяет нарисовать `n` линий уровня нормы векторного поля вместе с линиями тока.

StreamPlot[{$-1-x^3+y^2, 1+x^2-y^3$}, {$x,-3,3$}, {$y,-3,3$},
StreamPoints → **50**, **Mesh** → **15**] (*следующий рисунок слева *)

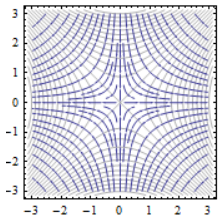
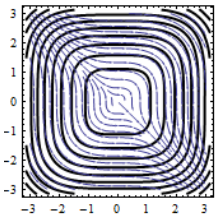
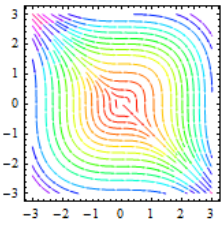
Можно задать конкретные значения этих линий уровня

StreamPlot[{$-1-x^3+y^2, 1+x^2-y^3$}, {$x,-3,3$}, {$y,-3,3$},
StreamPoints → **50**, **Mesh** → {{**1**, **2**, **3**, **5**, **10**, **20**}}]

(предыдущий рисунок справа).

По умолчанию строятся линии уровня величины/нормы векторного поля. Однако с помощью опции `MeshFunctions` можно определить скалярную функцию, линии уровня которой будут изображаться при использовании опции `Mesh`.

StreamPlot[{$x,-y$}, {$x,-3,3$}, {$y,-3,3$}, **Mesh** → **25**,
MeshFunctions−> **Function**[{x, y, **vx**, **vy**, n}, x^2-y^2]] (* рисунок слева *)

Опция MeshStyle управляет стилем этих линий уровня

StreamPlot[{$y^2,-x^2$}, {$x,-3,3$}, {$y,-3,3$}, **Mesh** → **10**,
MeshStyle−> {**Black**, **Thick**}]

(предыдущий рисунок в середине).

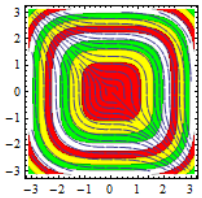
Величину/норму поля можно различать по цвету, используя опцию `StreamColorFunction`.

StreamPlot[{$y^2,-x^2$}, {$x,-3,3$}, {$y,-3,3$}, **StreamColorFunction** → **Hue**]

(предыдущий рисунок справа).

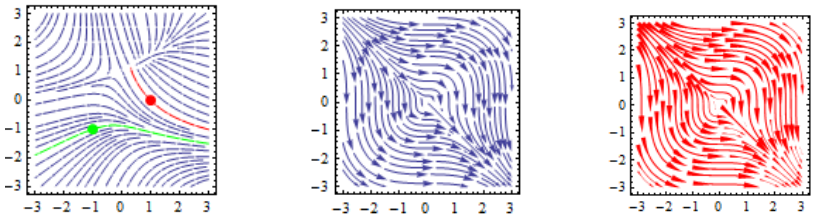
Области между линиями уровня можно закрашивать, используя опцию `MeshShading`.

StreamPlot[{$y^2, -x^2$}, {$x, -3, 3$}, {$y, -3, 3$}, **Mesh** → **10**,
MeshShading → {**Red**, **Yellow**, **Green**, **None**}]



Можно управлять стилем линий тока – толщиной, цветом, размером стрелочек и т.д. Некоторые линии тока можно выделить, указав для них отдельный стиль, например, цвет (следующий рисунок слева).

StreamPlot[{$-1 - x\text{^}2 + y, 1 + x - y\text{^}2$}, {$x, -3, 3$}, {$y, -3, 3$},
StreamPoints → {{{{**1**, **0**}, **Red**}, {{**−1**, **−1**}, **Green**}, **Automatic**}},
Epilog → {**Red**, **PointSize**[**Large**], **Point**[{**1**, **0**}], **Green**, **Point**[{**−1**, **−1**}]}]



Здесь для линий тока, которые проходят через точки (1,0) и (-1,-1), мы указали индивидуальный стиль (в нашем примере цвет), а для остальных линий – указали стиль Automatic.

Опция `StreamScale` управляет размером «наконечника» и длиной стрелок (предыдущий рисунок в середине).

StreamPlot[{$y^2, -x^2$}, {$x, -3, 3$}, {$y, -3, 3$}, **StreamScale** → **0.25**]

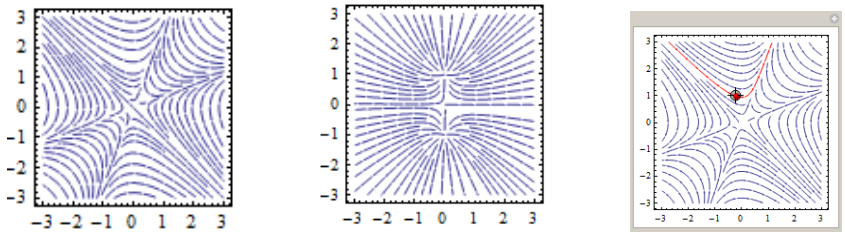
Опция `StreamStyle` управляет стилем линий тока

StreamPlot[{$y^2, -x^2$}, {$x, -3, 3$}, {$y, -3, 3$}, **StreamScale** → **0.4**,
StreamStyle → {"**Dart**", **Red**}] (* предыдущий рисунок справа *)

Для предварительного символьного вычисления выражения векторного поля необходимо использовать функцию `Evaluate`.

StreamPlot[**Evaluate**[**Grad**[$y^2x - x^2y$, {x, y}]], {$x, -3, 3$}, {$y, -3, 3$}]

(следующий рисунок слева). Попробуйте выполнить последнюю команду, опустив функцию `Evaluate`. Вы получите несколько сообщений и график построен не будет.



В качестве координат векторного поля можно использовать вещественные и мнимые части комплексных функций.

$z = x + Iy$;

StreamPlot[**Evaluate**@{**Re**[$z + 1/z$], **Im**[$z + 1/z$]}, {$x, -3, 3$}, {$y, -3, 3$}]

(предыдущий рисунок в середине).

Можно использовать динамические элементы системы для интерактивного слежения за линиями тока. Например,

```
Manipulate[ StreamPlot[ Evaluate[Grad[y^2 x - x^2 y, {x, y}]],
        {x, -3, 3}, {y, -3, 3}, StreamPoints -> {{{p, Red}, Automatic}},
        Epilog -> {Red, PointSize[Large], Point[p]}],
  {{p, {1, 0}}, Locator}]                    (* предыдущий рисунок справа *)
```

«Захватите» мышью «локатор» и перемещайте его. Каждое новое положение «локатора» задает новую точку и, следовательно, новую линию тока. Кривая будет динамически перерисовываться.

Можно рисовать линии тока и векторное поле совместно. Для этого нужно использовать опцию `VectorPoints`. Векторное поле изображается множеством векторов – стрелочек на плоскости, направление которых совпадает с направлением векторного поля, а длина в некотором масштабе равняется значению нормы векторного поля в соответствующих точках. Если `VectorPoints->n`, то в области графика на регулярной сетке точек `n x n` будут построены вектора поля.

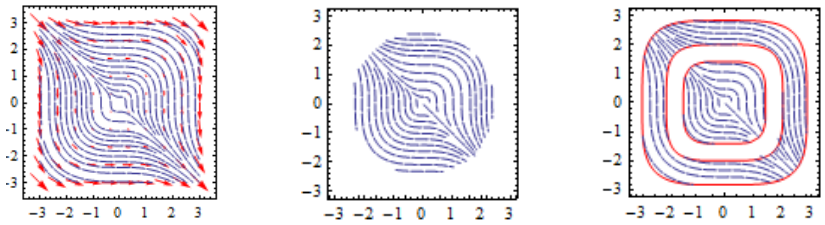
```
StreamPlot[{y^2, -x^2}, {x, -3, 3}, {y, -3, 3},
     VectorPoints -> 10, VectorStyle -> Red]
```

(следующий рисунок слева).

Область, в которой рисуются линии тока, можно ограничить, используя опцию `RegionFunction`

```
StreamPlot[{y^2, -x^2}, {x, -3, 3}, {y, -3, 3},
     RegionFunction -> Function[{x, y, vx, vy, n}, x^2 + y^2 < 6]]
```

(следующий рисунок в середине).

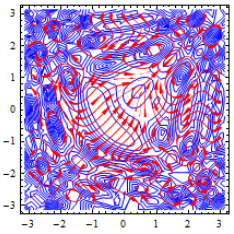
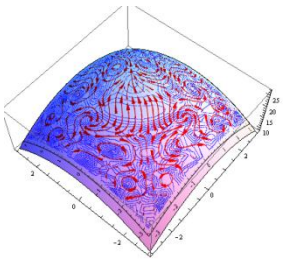
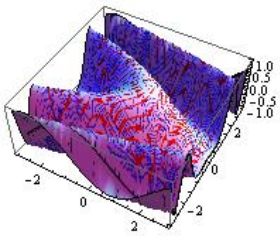


При этом ограничивающая область не обязана быть связной.

```
regionFn = Function[{x, y, vx, vy, n}, 4 < n < 8 || 0 < n < 2];
StreamPlot[{y^2, -x^2}, {x, -3, 3}, {y, -3, 3}, RegionFunction -> regionFn,
     BoundaryStyle -> Red]          (* предыдущий рисунок справа *)
```

Любопытна возможность использования графиков линий тока в качестве «поверхностных узоров». График линий тока, как и многие другие двумерные графические объекты, может быть наложен на поверхность как текстура.

```
strm = StreamPlot[{Cos[-1 - x + y^2], Sin[1 + x^2 - y]},
        {x, -3, 3}, {y, -3, 3}, StreamScale -> 0.25,
        StreamStyle -> {Dart, Red}, Mesh -> 10,
        MeshStyle-> {Blue}]          (* следующий рисунок слева *)
```

Используя директиву `Texture`, полученный «узор» накладывается на трехмерную поверхность

Plot3D[$27 - x^2 - y^2$, {x, −3, 3}, {y, −3, 3}, **Mesh** → **None**,
PlotStyle → **Texture**[**strm**]] (* предыдущий рисунок в середине *)

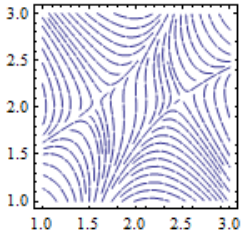
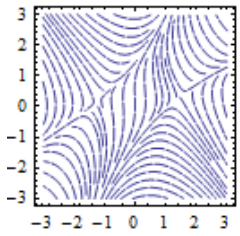
или на другую поверхность

Plot3D[**Sin**[$27 - x - y^2$], {x, −3, 3}, {y, −3, 3}, **Mesh** → **None**,
PlotStyle → **Texture**[**strm**]] (* предыдущий рисунок справа *)

Опции, которые связаны с оформлением графиков и которые есть у функции `StreamPlot`, очень похожи на те, которые имеются у большинства графических функций, в частности сходны с опциями функции `ContourPlot`. С большинством из них вы уже знакомы.

Если поле задано дискретно, т.е. задан набор векторов (пар чисел), то линии тока можно построить функцией `ListStreamPlot`. Она вначале выполняет интерполяцию переданных данных, а затем рисует линии тока, проходящие через регулярное множество точек в заданной области.

data = {{{−9, 7}, {0, −8}, {15, −20}}, {{9, 3}, {0, 2}, {6, 0}},
{{20, −27}, {2, −9}, {−6, 9}}};
ListStreamPlot[**data**] (* следующий рисунок слева *)

Обратите внимание на координаты, отложенные по осям графика. Они соответствуют индексам пар чисел (векторов). Если кроме векторов нужно указать точки их приложения, то список векторов надо дополнить координатами точек следующим образом.

data = {{{{−3, −3}, {−9, 7}}, {{−3, 0}, {0, −8}}, {{−3, 3}, {15, −20}}},
{{{0, −3}, {9, 3}}, {{0, 0}, {0, 2}}, {{0, 3}, {6, 0}}},
{{{3, −3}, {20, −27}}, {{3, 0}, {2, −9}}, {{3, 3}, {−6, 9}}}};
ListStreamPlot[**data**] (* предыдущий рисунок справа *)

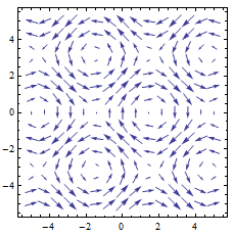
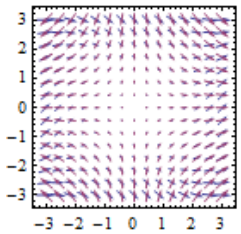
Например, первый вектор задан в виде {{-3,-3},{-9,7}} , где первая пара чисел представляет координаты точки, а вторая – координаты вектора. Общий вид поля не изменился, однако оно уже построено на реальных координатах точек.

Функция `ListStreamPlot` является аналогом функции `StreamPlot`. Во многом эти функции обладают идентичными возможностями и мы не будем здесь повторяться.

Другой способ визуализации двумерного векторного поля состоит в рисовании в плоской области стрелок, представляющих в некотором масштабе значение вектора $\vec{\mathbf{F}} = (P(x,y), Q(x,y))$ в этих точках. Такой рисунок в *Mathematica* создается функцией `VectorPlot`, которой в качестве аргументов передаются скалярные функции $P(x,y), Q(x,y)$.

В формате `VectorPlot[{P,Q},{x,xmin,xmax},{ymin,ymax}]` строится график векторного поля $(P(x,y), Q(x,y))$ на плоскости xy.

```
F[x_, y_] = {Sin[y], Cos[x]};
VectorPlot[F[x, y], {x, -5, 5}, {y, -5, 5}]   (* следующий рисунок слева *)
```

Допустимо построение нескольких векторных полей на одном графике.

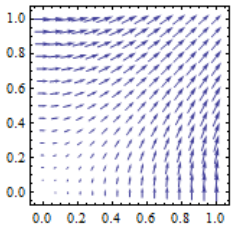
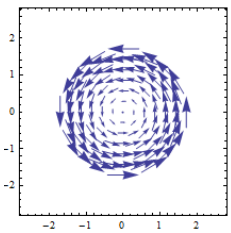
```
VectorPlot[{{x^2 + y^2, x^2 - y^2}, {x, -y}}, {x, -3, 3}, {y, -3, 3}]
```

(предыдущий рисунок справа).

Используйте функцию `Evaluate` для символьного вычисления векторного поля перед его построением.

```
VectorPlot[Evaluate[Grad[Sin[xy], {x, y}]], {x, 0, 1}, {y, 0, 1}]
```

(следующий рисунок слева).

Попробуйте выполнить предыдущую команду без функции `Evaluate` – ничего не выйдет.

Область, в которой рисуется векторное поле, можно ограничить, используя опцию `RegionFunction`

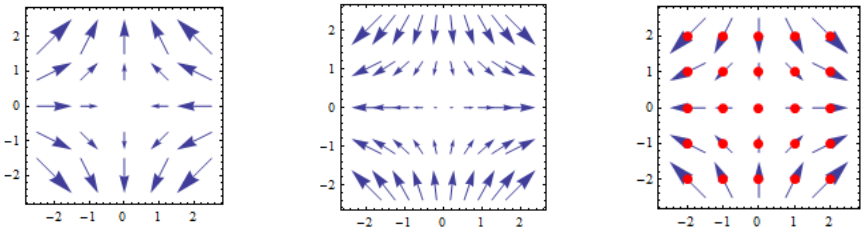
```
VectorPlot[{-y, x}, {x, -2, 2}, {y, -2, 2}, VectorScale -> .25,
    RegionFunction -> Function[{x, y, vx, vy, n}, x^2 + y^2 <= 3]]
```

(предыдущий рисунок справа).

Можно управлять количеством стрелок на графике с помощью опции `VectorPoints→n`. По умолчанию функция `VectorPlot` рисует вектора на регулярной сетке точек `n x n`.

```
VectorPlot[{-x, y}, {x, -2, 2}, {y, -2, 2}, VectorPoints -> 5]
```

(следующий рисунок слева)



Количество точек по разным осям можно задавать различным.

VectorPlot[{x, $-y$}, {x, -2, 2}, {y, -2, 2}, **VectorPoints** → {**10**, **5**}]

(предыдущий рисунок в середине).

Точки, в которых будут рисоваться стрелочки, можно задавать самостоятельно.

points = **Flatten**[**Table**[{x, y}, {x, -2, 2, 1}, {y, -2, 2, 1}], 1]

VectorPlot[{x, $-y$}, {x, -2, 2}, {y, -2, 2}, **VectorPoints** → **points**,

VectorScale →. **25**, **Epilog** → {**Red**, **PointSize**[**Large**], **Point**[**points**]}]

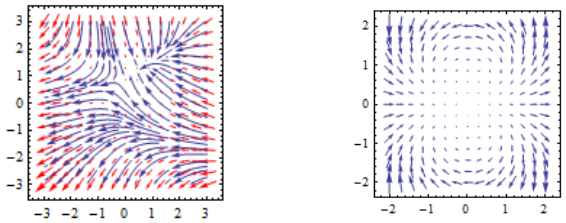
(предыдущий рисунок справа). Здесь опция `VectorScale` управляет размером стрелок.

Опция `StreamPoints->n` позволяет вместе с векторным полем нарисовать `n` линий тока.

VectorPlot[{$-1 - x\text{^}2 + y$, $1 + x - y\text{^}2$}, {x, -3, 3}, {y, -3, 3},

VectorStyle → **Red**, **StreamPoints** → **50**,

VectorScale →. **1**, **StreamScale** → **0**. **2**] (* следующий рисунок слева *)



Можно использовать комплексные функции

$z = x + y\,i$;

VectorPlot[{**Re**[z^2], **Im**[z^2]}, {x, -2, 2}, {y, -2, 2}, **VectorScale** →. **1**]

(предыдущий рисунок справа).

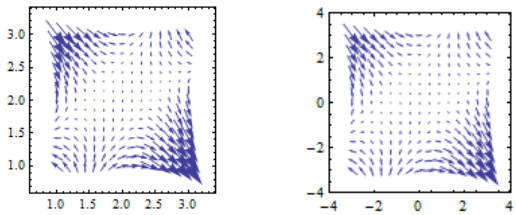
Если поле задано дискретно, т.е. задан набор векторов (пар чисел), то векторное поле можно построить функцией `ListVectorPlot`. Вначале она выполняет интерполяцию переданных данных, а затем рисует стрелки векторов на регулярном множестве точек области.

data = {{{-9, 7}, {0, -8}, {**15**, -20}},

{{**9**, **3**}, {**0**, **2**}, {**6**, **0**}},

{{**20**, -27}, {**2**, -9}, {-6, **9**}}};

ListVectorPlot[**data**, **VectorScale** → **0**. **25**] (* следующий рисунок слева *)



Здесь опция `VectorScale` задает относительную длину стрелок.

Если кроме векторов нужно задать точки их приложения, то список векторов надо дополнить координатами точек следующим образом.

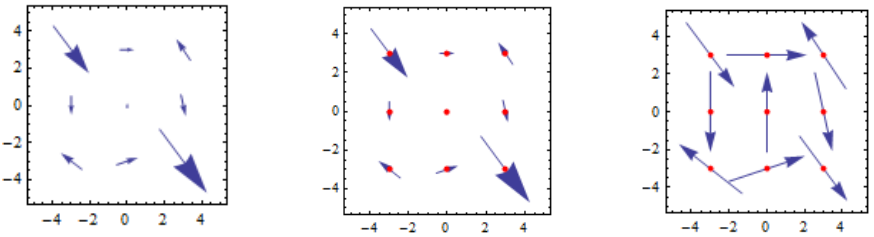
```
data = {{{{-3, -3}, {-9, 7}}, {{-3, 0}, {0, -8}}, {{-3, 3}, {15, -20}}},
        {{{0, -3}, {9, 3}}, {{0, 0}, {0, 2}}, {{0, 3}, {6, 0}}},
        {{{3, -3}, {20, -27}}, {{3, 0}, {2, -9}}, {{3, 3}, {-6, 9}}}};
ListVectorPlot[data, VectorScale → 0.2]  (* предыдущий рисунок справа *)
```

Графики на предыдущем рисунке отличаются только значениями, указанными на координатных осях.

Можно изобразить только те вектора, которые были преданы. Для этого используется значение опции `VectorPoints→All`.

```
ListVectorPlot[data, VectorScale → 0.5, VectorPoints → All]
```

(следующий рисунок слева).



Можно изобразить вектора вместе с точками их приложения

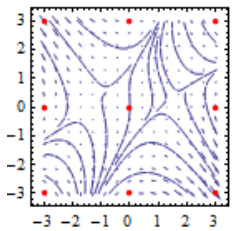
```
points = Flatten[Table[data[[j, i, 1]], {j, 3}, {i, 3}], 1];
ListVectorPlot[data, VectorScale → 0.5, VectorPoints → All,
  Epilog → {Red, PointSize[0.03], Point[points]}] (* пред. рис. в середине *)
```

Можно рисовать стрелочки постоянного размера (те же значения **data** и **points**, что и в предыдущем коде).

```
ListVectorPlot[data, VectorScale → {0.5, 0.5, None}, VectorPoints → All,
  Epilog → {Red, PointSize[0.03], Point[points]}]   (* пред. рис. справа *)
```

Вместе с векторным полем можно рисовать линии тока поля. Для этого используется опция `StreamPoints`.

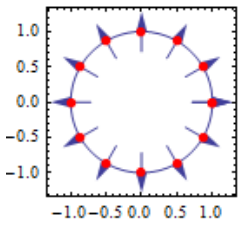
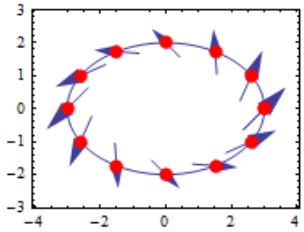
```
points = Flatten[Table[data[[j, i, 1]], {j, 3}, {i, 3}], 1];
ListVectorPlot[data, Epilog → {Red, PointSize[0.03], Point[points]},
               StreamPoints → 20]
```



Пример. Нарисовать поле нормалей единичной окружности.

```
p = {x, y};
v = {x, y};                          (* векторное поле *)
r[t_] = {Cos[t], Sin[t]};            (* параметрическое уравнение кривой *)
dt = {p, v} /. Thread[p → r[t]];
data = Table[dt, {t, 0, 2π, π/6}];   (* точки вместе с векторами *)
```

```
points = Table[data[[i, 1]], {i, Length[data]}];        (* только точки *)
lvp = ListVectorPlot[data, VectorPoints → All, VectorScale → 0.1,
                     Epilog → {Red, PointSize[0.02], Point[points]}];
ln = ParametricPlot[r[t], {t, 0, 2π}];
Show[lvp, ln]                                          (* следующий рисунок слева *)
```

Если вы измените вторую и третью строки предыдущего кода, например так

```
v = {x − y, x + y};
r[t_] = {3Cos[t], 2Sin[t]};
```

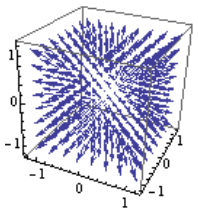
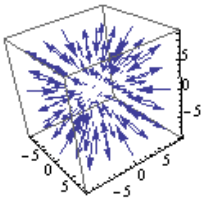
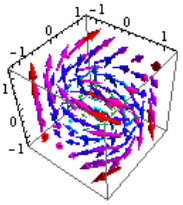
то построите другое векторное поле на другой кривой (предыдущий рисунок справа).

□

Многие опции функций `VectorPlot` и `ListVectorPlot` имеют такой же смысл как и у функции `Graphics` и мы не будем здесь повторяться, поясняя их значение.

Для построения векторных полей в трехмерном пространстве используется функция `VectorPlot3D`.

```
VectorPlot3D[{x, y, z}, {x, −1, 1}, {y, −1, 1}, {z, −1, 1}] (* след. рисунок слева *)
```

Используйте функцию `Evaluate` для символьного вычисления векторного поля перед его построением.

```
VectorPlot3D[Evaluate[D[Sin[x^2 + y^2 + z^2], {{x, y, z}}]],
    {x, −2π, 2π}, {y, −2π, 2π}, {z, −2π, 2π}, VectorPoints → 5]
```

(предыдущий рисунок в середине).

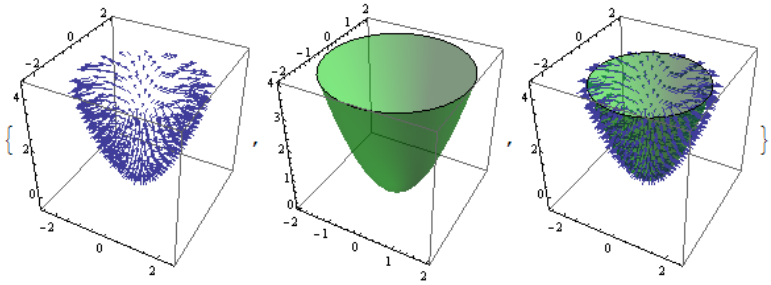
Можно управлять цветом и формой стрелок. В следующей команде форма стрелок задается опцией `VectorStyle→"Arrow3D"`, а опция `VectorColorFunction→Hue` задает цвет в соответствии с нормой (`Norm`) векторного поля.

```
VectorPlot3D[{y, −x, z}, {x, −1, 1}, {y, −1, 1}, {z, −1, 1},
    PlotRange → All, VectorPoints → 5, VectorColorFunction → Hue,
    VectorStyle → "Arrow3D"]        (* предыдущий рисунок справа *).
```

В следующем примере мы рисуем поле градиента скалярного поля $u = x^2 + y^2 - z$ в окрестности поверхности $u = 0$. Для этого мы используем

опцию `RegionFunction`. С ее помощью мы выделяем узкую область векторного поля в окрестности этой поверхности.

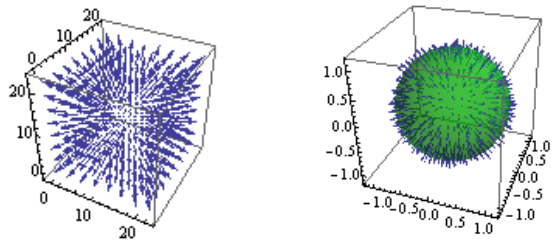
```
scalarField = x^2 + y^2 - z;
vectorField = D[scalarField, {{x, y, z}}];
v = VectorPlot3D[vectorField, {x, -2, 2}, {y, -2, 2}, {z, 0, 4},
      VectorPoints → 25, VectorScale → {0.15, Scaled[0.5]},
      RegionFunction → Function[{x, y, z}, -0.1 ≤ scalarField <= 0.1]];
c = ContourPlot3D[scalarField == 0, {x, -2, 2}, {y, -2, 2}, {z, 0, 4},
      Mesh → None, ContourStyle → Opacity[0.5, Green]];
{v, c, Show[v, c]}
```



Если векторное поле задано дискретно, т.е. задан набор векторов (троек чисел), то векторное поле можно построить функцией `ListVectorPlot3D`. Как и другие аналогичные функции она вначале выполняет интерполяцию данных, а потом рисует стрелки векторов.

```
ListVectorPlot3D[Table[{x, y, z}, {x, -1, 1, .1}, {y, -1, 1, .1}, {z, -1, 1, .1}]]
```

(следующий рисунок слева).



Обратите внимание на координаты точек векторного поля. Они представляются индексами векторов {*x*,*y*,*z*}, а не значениями *x*,*y*,*z*.

В следующем коде мы изображаем векторное поле нормалей к сфере. Но с помощью опции `DataRange` это поле отнесено к реальным координатам точек, а не к индексам векторов.

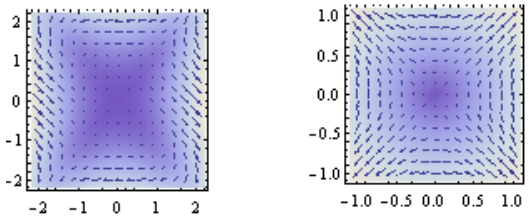
```
scalarField = x^2 + y^2 + z^2 - 1;
vectorField = D[scalarField, {{x, y, z}}];
data = Table[vectorField, {x, -1, 1}, {y, -1, 1}, {z, -1, 1}];
v = ListVectorPlot3D[data, DataRange → {{-1, 1}, {-1, 1}, {-1, 1}},
      VectorPoints → 15, VectorScale → {0.1, Scaled[0.5]},
      RegionFunction → Function[{x, y, z}, -0.1 ≤ scalarField <= 0.1]];
c = ContourPlot3D[scalarField == 0, {x, -1, 1}, {y, -1, 1}, {z, -1, 1},
      Mesh → None, ContourStyle → Opacity[0.5, Green]];
Show[v, c]   (* предыдущий рисунок справа *)
```

Для одновременной визуализации скалярного и векторного поля можно использовать функции `VectorDensityPlot` и `StreamDensityPlot`. Они строят график векторного поля или его линий тока на фоне графика плотности нормы этого поля (или другой скалярной функции, которую укажет пользователь). Фон раскрашивается в цвета, яркость которых соответствует значению этого скалярного поля. В формате

`VectorDensityPlot[{vx,vy},{x,xmin,xmax},{ymin,ymax}]`

рисуются стрелочки векторного поля, а скалярной функцией, значения которой используются для графика плотности, является норма векторного поля `{vx,vy}`.

```
VectorDensityPlot[{x^2 - y^2, -x^2}, {x, -2, 2}, {y, -2, 2}]   (* след. рис. слева *)
```



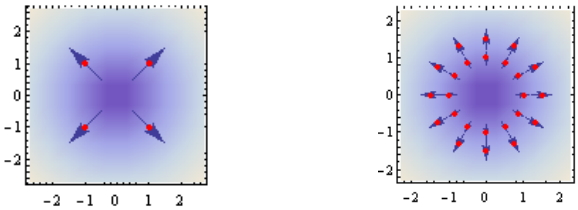
Используйте функцию `Evaluate` для символьного вычисления векторного поля перед его построением

```
VectorDensityPlot[Evaluate[D[x y, {{x, y}}]], {x, -1, 1}, {y, -1, 1}]
```

(предыдущий рисунок справа).

Можно явно указать точки, в которых должны быть нарисованы стрелочки. Для этого используется опция `VectorPointe→точки`.

```
points = {{-1, -1}, {-1, 1}, {1, -1}, {1, 1}};
VectorDensityPlot[{x, y}, {x, -2, 2}, {y, -2, 2},
    VectorPoints → points, VectorScale →.25,
    Epilog → {Red, PointSize[Medium], Point[points]}] (* след. рис. слева *)
```



В качестве примера изобразим вектора поля только на двух окружностях

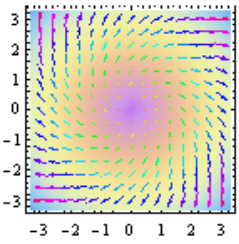
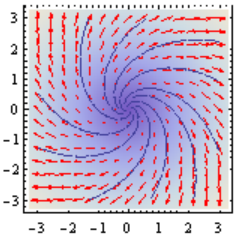
```
points = Join[Table[{Cos[t], Sin[t]}, {t, 0, 2π, π/6}],
              Table[{1.5Cos[t], 1.5Sin[t]}, {t, 0, 2π, π/6}]];
VectorDensityPlot[{x, y}, {x, -2, 2}, {y, -2, 2},
              VectorPoints → points, VectorScale → Large,
              Epilog → {Red, PointSize[Medium], Point[points]}]
```

(предыдущий рисунок справа).

Опции `ColorFunction` и `VectorColorFunction` позволяют управлять цветом фона и стрелочек. В следующем примере значением, которое используется для выбора цвета является норма векторного поля.

```
VectorDensityPlot[{y + x, y - x}, {x, -3, 3}, {y, -3, 3},
        ColorFunction → "Pastel", VectorColorFunction → Hue]
```

(следующий рисунок слева).

Можно рисовать векторное поле и линии его тока совместно. Для этого используется опция `StreamPoints→n`, где `n` количество линий тока.

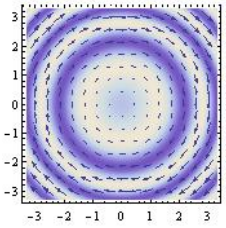
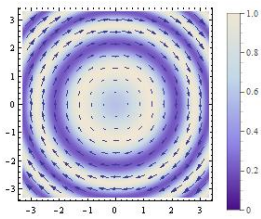
```
VectorDensityPlot[{x + y, y - x}, {x, -3, 3}, {y, -3, 3},
        VectorStyle → Red, StreamPoints → 10]  (* пред. рис. справа *)
```

В формате

```
VectorDensityPlot[{{vx, vy}, s}, {x, xmin, xmax}, {ymin, ymax}]
```

выражение `s` определяет скалярную функцию, значения которой будут использоваться при построении графика плотности (фона).

```
VectorDensityPlot[{{y, -x}, Sin[x^2 + y^2]}, {x, -3, 3}, {y, -3, 3},
        MaxRecursion → 3]        (* следующий рисунок слева *)
```

У функции `VectorDensityPlot` имеется возможность построения шкалы цветов.

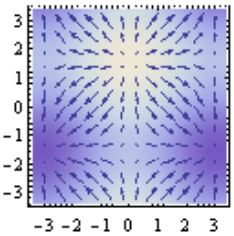
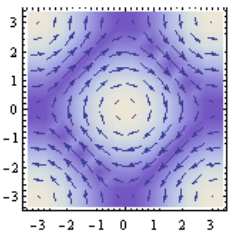
```
VectorDensityPlot[{{y, -x}, Sin[x^2 + y^2]}, {x, -3, 3}, {y, -3, 3},
    MaxRecursion → 3, PlotLegends-> BarLegend[{"LakeColors", {0, 1}}]]
```

(предыдущий рисунок справа).

Построим векторное поле на фоне его дивергенции

```
f = {Sin[x], -Cos[y]};
s = Div[f, {x, y}]
VectorDensityPlot[{f, s}, {x, -3, 3}, {y, -3, 3},
        VectorPoints → 12]        (* следующий рисунок слева *)
```

$\mathrm{Cos}[x] + \mathrm{Sin}[y]$

Построим векторное поле на фоне нормы его ротора

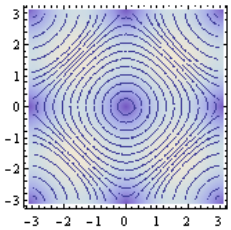
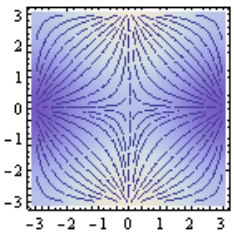
```
f = {Sin[y], -Sin[x]};
crl = Curl[f, {x, y}]
VectorDensityPlot[{f, Norm[crl]}, {x, -3, 3}, {y, -3, 3},
        VectorPoints → 12]        (* предыдущий рисунок справа *)
```

$-\mathrm{Cos}[x] - \mathrm{Cos}[y]$

Функция StreamDensityPlot строит линии тока на фоне графика плотности некоторого скалярного поля. По умолчанию этим скалярным полем является норма векторного поля.

StreamDensityPlot[{**Sin**[y], −**Sin**[x]}, {x, −3, 3}, {y, −3, 3}]

(следующий рисунок слева)

В формате

StreamDensityPlot[{{v_x, v_y}, s}, {x, x_{min}, x_{max}}, {y_{min}, y_{max}}]

выражение s задает скалярную функцию, значения которой будут использоваться при построении графика плотности. В качестве примера построим линии тока поля на фоне его дивергенции.

f = {**Sin**[x], −**Sin**[y]};

s = **Div**[f, {x, y}]

StreamDensityPlot[{f, s}, {x, −3, 3}, {y, −3, 3}] (* предыдущий рис. справа *)

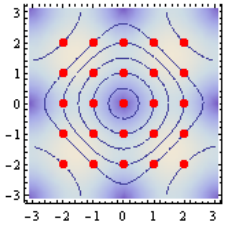
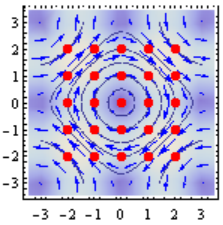
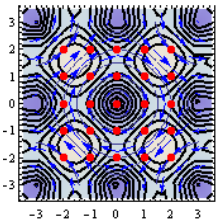
Cos[x] − Cos[y]

Построим линии тока, которые проходят через заданные точки. Для этого используем опцию StreamPoints→точки.

points = **Tuples**[{−2, −1, 0, 1, 2}, 2];

StreamDensityPlot[{**Sin**[y], −**Sin**[x]}, {x, −3, 3}, {y, −3, 3},
StreamPoints → **points**, **Epilog** → {**Red**, **PointSize**[0.02], **Point**[**points**]}]

(следующий рисунок слева)

К графику линий тока можно добавить график векторного поля (стрелочки).

StreamDensityPlot[{**Sin**[y], −**Sin**[x]}, {x, −3, 3}, {y, −3, 3},
StreamPoints → **points**,
Epilog → {**Red**, **PointSize**[0.04], **Point**[**points**]},
VectorPoints → 10, **VectorStyle** → **Blue**] (* пред. рис. в середине *)

Имеется также опция Mesh, которая позволяет рисовать линии уровня некоторой скалярной функции. По умолчанию такой функцией является норма векторного поля.

StreamDensityPlot[{**Sin**[y], −**Sin**[x]}, {x, −3, 3}, {y, −3, 3},
StreamPoints → **points**, **VectorPoints** → 10, **VectorStyle** → **Blue**,
Epilog → {**Red**, **PointSize**[0.04], **Point**[**points**]},
Mesh → 10, **MeshStyle** → **Thick**] (* предыдущий рисунок справа *)

Для дискретных наборов данных аналогами функций VectorDensityPlot и StreamDensityPlot являются функции ListVectorDensityPlot и ListStreamDensityPlot. Возможности этих функций сходны и мы не будем на них останавливаться.

3.5 Ряды

3.5.1 Вычисление конечных и бесконечных сумм.

Для символьного вычисления конечных и бесконечных сумм используется функция Sum. В формате Sum[f,{i,i_{max}}] она вычисляет сумму вида $\sum_{i=1}^{i_{max}} f$.

Sum[$2k-1$,{k,**12**}]

144

Sum[f,{i,i_{max}}] может быть введена как $\sum_{i}^{i_{max}} f$. Трафарет ввода суммы можно найти на панели специальных символов «Basic Math Input». Можно также использовать специальные комбинации клавиш. Для ввода знака суммы» наберите Esc–sum–Esc. Потом надо ввести подстрочный и надстрочный индексы у значка суммы Σ. Для этого введите Ctrl–_ (подчеркивание), затем Ctrl–% (или Ctrl–^, затем Ctrl–%). Введите значения нижнего и верхнего пределов суммирования. Ctrl–пробел возвращает курсор на нормальный уровень ввода, где введите суммируемое выражение. Следующая команда эквивалентна предыдущей.

$\sum_{\mathbf{k}}^{\mathbf{12}} (\mathbf{2k-1})$

144

В формате Sum[f,{i,i_{min},i_{max}}] задается начальное $i = i_{min}$ и конечное i_{max} значение счетчика.

Sum[k,{k,**4**,**8**}]

30

или

$\sum_{\mathbf{k=4}}^{\mathbf{8}} \mathbf{k}$

30

В формате Sum[f,{i,i_{min},i_{max},di}] задается еще шаг счетчика di.

Sum[k,{k,**4**,**12**,**3**}]

21

В формате Sum[f,{i,i_{min},i_{max}},{j_{min},j_{max}}] вычисляется двукратная сумма вида $\sum_{i=i_{min}}^{i_{max}} \sum_{j=j_{min}}^{j_{max}} f$.

Sum[$(i+j)^2$,{i,**3**},{j,**4**}]

266

Допустимо суммирование по большему количеству переменных/индексов.

Можно суммировать выражения, содержащие более чем одну переменную; другие переменные будут трактоваться, как константы.

```
Sum[x^k/k!, {k, 1, 7}]
```

$$x+\frac{x^2}{2}+\frac{x^3}{6}+\frac{x^4}{24}+\frac{x^5}{120}+\frac{x^6}{720}+\frac{x^7}{5040}$$

Как и выше можно задать шаг изменения переменной суммирования.

```
Sum[x^k/k!, {k, 1, 7, 2}]
```

$$x+\frac{x^3}{6}+\frac{x^5}{120}+\frac{x^7}{5040}$$

В двойной сумме «внутренние» пределы суммирования могут зависеть от внешнего индекса.

```
Sum[Sin[x^m + y^n], {m, 3}, {n, m}]
```

$$\text{Sin}[x+y]+\text{Sin}[x^2+y]+\text{Sin}[x^3+y]+\text{Sin}[x^2+y^2]+\text{Sin}[x^3+y^2]+\text{Sin}[x^3+y^3]$$

Результат вычисления суммы может быть представлен символически.

```
Sum[k^4, {k, n}]
```

$$\frac{1}{30}n(1+n)(1+2n)(-1+3n+3n^2)$$

```
Sum[(-1)^k k/(4k^2 - 1), {k, n}]
```

$$\frac{-1+(-1)^n-2n}{4(1+2n)}$$

```
Sum[1/(i(i + 1)(i + 2)), {i, n}]
```

$$\frac{1}{4}-\frac{1}{2(1+n)(2+n)}$$

Пределы суммирования могут быть бесконечными (`Infinity`). Такие суммы принято называть рядами и *Mathematica* умеет их вычислять.

```
Sum[1/k^2, {k, ∞}]
```

$$\frac{\pi^2}{6}$$

$$\sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{(2k-1)^5}$$

$$\frac{5\pi^5}{1536}$$

$$\sum_{k=1}^{\infty} \frac{(-1)^{k-1}k}{(k+1)^2}$$

$$\frac{\pi^2}{12} - \text{Log}[2]$$

$$\sum_{k=0}^{\infty} \frac{1}{k!\,(k+z)!}$$

$$\text{BesselI}[z, 2]$$

Множество, которое пробегает индекс суммирования, можно задавать конечным списком.

```
Sum[f[i], {i, {a, b, c}}]
```

$$f[a] + f[b] + f[c]$$

Если найти результат суммирования в символьной форме не удается, *Mathematica* возвращает сумму невычисленной.

```
Sum[n/(n! + 1), {n, Infinity}]
```

$$\sum_{n}^{\infty} \frac{n}{1+n!}$$

Однако, для числовых рядов можно получить приближенное значение суммы, используя функцию `N`.

```
N[%]
```

```
1.8064971
```

Применение функции `N` к результату, полученному с помощью функции `Sum`, эквивалентно использованию функции `NSum`, которая служит для численного суммирования рядов. Для бесконечных числовых рядов результат получается только в случае, если ряд сходится.

```
NSum[n/(n! + 1), {n, Infinity}]
```

```
1.8064971
```

```
ns = NSum[(-3)^i/i!, {i, 0, ∞}]
```

```
0.049787068
```

Последняя сумма вычисляется символьно

```
ts = Sum[(-3)^i/i!, {i, 0, ∞}]
```

$$\frac{1}{e^3}$$

и ошибка вычислений составляет

```
ns - ts
```

```
3.469447×10^-17
```

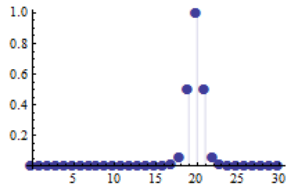
У функции `NSum` имеются опции, которые управляют точностью вычислений. `AccuracyGoal` определяет абсолютную погрешность вычислений, используя

количество значащих цифр. `PrecisionGoal->n` определяет для значения *x* погрешность вычислений равную $|x|10^{-n}$. Когда включены опции `AccuracyGoal->m` и `PrecisionGoal->n`, то *Mathematica* пытается выполнять вычисления величины *x* с погрешностью $\max\left(10^{-m}, |x|10^{-n}\right)$. Опция `WorkingPrecision` определяет количество значащих цифр, используемых во внутренних вычислениях; установка `WorkingPrecision→ MachinePrecision` приводит к вычислениям с процессорной точностью. Использование этих опций идентично их использованию для функции `NIntegrate`.

```
ns = NSum[(-3)^i/i!, {i, 0, ∞}, WorkingPrecision → 4]
0.04979
```

По умолчанию функция `NSum` суммирует первые 15 членов перед тем, как переходит к аппроксимации «хвоста» ряда. Однако это не всегда корректно. Например, построим график следующей последовательности

```
DiscretePlot[1/((k - 20)^4 + 1), {k, 0, 30}, PlotRange → All,
                                 PlotMarkers → {Automatic, Medium}]
```



Максимальное значение последовательность достигает на $20-$м члене. Поэтому суммирование $15-$ти членов с последующей аппроксимацией «хвоста» будет иметь большую погрешность

```
NSum[1/((k - 20)^4 + 1), {k, 0, ∞}]
2.2214029
```

Чтобы вначале просуммировать большее количество элементов используется опция `NSumTerms`.

```
NSum[1/((k - 20)^4 + 1), {k, 0, ∞}, NSumTerms → 30]
2.1569165
```

По умолчанию функция `NSum` выполняет проверку сходимости ряда. Если вы заранее знаете, что ряд сходится, то отключив проверку сходимости опцией `VerifyConvergence→False` можно ускорить вычисления.

Если ряд не сходится, то функция `NSum` в качестве результата возвращает `ComplexInfinity`.

```
NSum[1, {k, 1, ∞}]
ComplexInfinity
```

Но часто она выводит «угрожающие» сообщения, и представляет какой – то численный результат, которому нельзя доверять.

$\mathbf{NSum}[\mathbf{1}/\sqrt{\boldsymbol{k}},\{\boldsymbol{k},\mathbf{1},\infty\}]$

```
Numerical integration converging too slowly…
```

$3.8439417962027 54\times 10^{13977}$

В *Mathematica* имеются функции `Product` и `NProduct`, вычисляющие произведения $\prod_{i=i_{\min}}^{i_{\max}} f$. Обращение к ним аналогично формату обращения к функциям `Sum` и `NSum`.

$\mathbf{Product}[\boldsymbol{i},\{\boldsymbol{i},\boldsymbol{n}\}]$

$n!$

$\boldsymbol{p}=\mathbf{4};$

$$\mathbf{HoldForm}\left[\mathbf{Sum}\left[\frac{\mathbf{1}}{\prod_{\boldsymbol{i}=\mathbf{0}}^{\boldsymbol{p}}(\boldsymbol{\alpha}+\boldsymbol{n}+\boldsymbol{i})},\{\boldsymbol{n},\mathbf{1},\infty\}\right]\right]==\mathbf{Sum}[\frac{\mathbf{1}}{\prod_{\boldsymbol{i}=\mathbf{0}}^{\boldsymbol{p}}(\boldsymbol{\alpha}+\boldsymbol{n}+\boldsymbol{i})},\{\boldsymbol{n},\mathbf{1},\infty\}]$$

$$\sum_{n=1}^{\infty}\frac{1}{\prod_{i=0}^{p}(\alpha+n+i)}==\frac{1}{4(1+\alpha)(2+\alpha)(3+\alpha)(4+\alpha)}$$

Здесь функция `HoldForm` используется для того, чтобы в левой части записать выражение в невычисленном виде.

Часто результат символьного суммирования зависит от значений параметров, стоящих в суммируемом выражении. Если при каких – то значениях параметров ряд сходится, то возвращается результат, соответствующий таким значениям. Если вы хотите увидеть условия, при которых этот результат верен, то следует использовать опцию `GenerateConditions→True`.

$\mathbf{Sum}[\boldsymbol{a}^{\boldsymbol{n}},\{\boldsymbol{n},\mathbf{0},\infty\},\mathbf{GenerateConditions}\rightarrow\mathbf{True}]$

$\text{ConditionalExpression}[\frac{1}{1-a},\text{Abs}[a]<1]$

Если в полученном результате вы желаете избавиться от предположений, то используйте функцию `Refine`.

$\mathbf{Refine}[\%,\mathbf{Abs}[\boldsymbol{a}]<1]$

$$\frac{1}{1-a}$$

Если опцию `GenerateConditions→True` не использовать, то сразу получим результат (без предположений, хотя они использовались при вычислении).

$\mathbf{Sum}[\boldsymbol{a}^{\boldsymbol{n}},\{\boldsymbol{n},\mathbf{0},\infty\}]$

$$\frac{1}{1-a}$$

Разложение не всегда получается таким, как хочется

$\mathbf{sm}=\mathbf{Sum}[\frac{\mathbf{Sin}[\boldsymbol{n}\,\boldsymbol{x}]}{\boldsymbol{n}},\{\boldsymbol{n},\mathbf{1},\infty\}]$

$\frac{1}{2}\,i(\text{Log}[1-e^{ix}]-\text{Log}[e^{-ix}(-1+e^{ix})])$

Для получения более простого выражения следует выполнить алгебраические преобразования результата

$$\mathbf{logrule} = \mathbf{Log}[\mathbf{a\_}] - \mathbf{Log}[\mathbf{b\_}] \to \mathbf{Log}[\boldsymbol{a/b}];$$
$$\mathbf{sm1} = \mathbf{Simplify}[\mathbf{sm}/.\mathbf{logrule}]$$
$$\mathbf{PowerExpand}[\mathbf{sm1}, \mathbf{Assumptions} :\to \mathbf{0} < \boldsymbol{x} < 2\boldsymbol{\pi}]$$
$$\frac{\pi}{2} - \frac{x}{2}$$

В выражении **sm** мы выполнили подстановку Log[*a*_]–Log[*b*_]→Log[*a*/*b*], а следом использовали функцию `PowerExpand` для выполнения преобразования `Log[a^b]` в `bLog[a]`, которое автоматически функцией `Simplify` не выполняется. Т.о. имеет место равенство $\sum_{n=1}^{\infty} \frac{\sin n x}{n} = \frac{\pi - x}{2}$.

3.5.2 Ряды Тейлора.

Произвольную функцию `f(x)` можно разложить в степенной ряд около точки $x=x_0$, используя функцию `Series`. В формате `Series[f,{x,x0,n}]` она генерирует степенной ряд $\sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k$ в окрестности точки x_0 до слагаемого $(x\text{-}x_0)^n$. Например,

$$\mathbf{Series}[\boldsymbol{f}[\boldsymbol{x}], \{\boldsymbol{x}, \mathbf{0}, \mathbf{5}\}]$$
$$f[0] + f'[0]x + \frac{1}{2}f''[0]x^2 + \frac{1}{6}f^{(3)}[0]x^3 + \frac{1}{24}f^{(4)}[0]x^4 + \frac{1}{120}f^{(5)}[0]x^5 + O[x]^6$$
$$\mathbf{g} = \mathbf{Series}[\mathbf{Exp}[\boldsymbol{x}], \{\boldsymbol{x}, \mathbf{0}, \mathbf{5}\}]$$
$$1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + O[x]^6$$
$$\mathbf{Series}[\frac{\mathbf{Sin}[\boldsymbol{x}]}{\boldsymbol{x}}, \{\boldsymbol{x}, \mathbf{0}, \mathbf{8}\}]$$
$$1 - \frac{x^2}{6} + \frac{x^4}{120} - \frac{x^6}{5040} + \frac{x^8}{362880} + O[x]^9$$

Как видите, в конце полинома, представляющего ряд, функция `Series` выводит символ $O\left[x^{n+1}\right]$. Следует помнить, что функция `Series` создает ненулевой остаточный член, даже если он тождественно равен нулю.

$$\mathbf{Series}[\boldsymbol{x}^{\mathbf{5}}, \{\boldsymbol{x}, \mathbf{0}, \mathbf{5}\}]$$
$$x^5 + O[x]^6$$
$$\boldsymbol{P}[\mathbf{x\_}] = \boldsymbol{x}^{\mathbf{5}} + \mathbf{2}\boldsymbol{x}^{\mathbf{4}} - \mathbf{3}\boldsymbol{x}^{\mathbf{3}} + \mathbf{6}\boldsymbol{x}^{\mathbf{2}} - \mathbf{12}\boldsymbol{x} + \mathbf{17};$$
$$\mathbf{Series}[\boldsymbol{P}[\boldsymbol{x}], \{\boldsymbol{x}, \mathbf{0}, \mathbf{5}\}]$$
$$17 - 12x + 6x^2 - 3x^3 + 2x^4 + x^5 + O[x]^6$$

Функцию `Series` можно использовать для переразложеня полинома по степеням $x - a$ при любом a.

$$\mathbf{Series}[\boldsymbol{P}[\boldsymbol{x}], \{\boldsymbol{x}, \mathbf{1}, \mathbf{5}\}]$$
$$11 + 4(x-1) + 19(x-1)^2 + 15(x-1)^3 + 7(x-1)^4 + (x-1)^5 + O[x-1]^6$$

С рядами можно выполнять обычные арифметические операции.

$\mathbf{G = g\text{^}2(1 + g)}$ (* **g** это разложение в ряд экспоненты, см. выше *)

$$2 + 5x + \frac{13x^2}{2} + \frac{35x^3}{6} + \frac{97x^4}{24} + \frac{55x^5}{24} + O[x]^6$$

$$\frac{\mathbf{Series[Log[1 + x], \{x, 0, 7\}]}}{\mathbf{Series[Log[1 + x^2], \{x, 0, 7\}]}}$$

$$\frac{1}{x} - \frac{1}{2} + \frac{5x}{6} - \frac{x^2}{2} + \frac{17x^3}{60} - \frac{x^4}{4} + O[x]^5$$

$\boldsymbol{t = Series[ArcTan[x], \{x, 0, 8\}]}$

$$x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + O[x]^9$$

t^2

$$x^2 - \frac{2x^4}{3} + \frac{23x^6}{45} - \frac{44x^8}{105} + O[x]^{10}$$

Можно разлагать в степенные ряды функции нескольких переменных

$\mathbf{Series[Sin[xy], \{x, 0, 5\}, \{y, 0, 5\}]}$

$$(y + O[y]^6)x + (-\frac{y^3}{6} + O[y]^6)x^3 + (\frac{y^5}{120} + O[y]^6)x^5 + O[x]^6$$

Функция `Series` создает объект `SeriesData`, который можно использовать как функцию.

$\mathbf{scos = Series[Cos[x], \{x, 0, 5\}]}$

$$1 - \frac{x^2}{2} + \frac{x^4}{24} + O[x]^6$$

$\mathbf{scos * x^2}$

$$x^2 - \frac{x^4}{2} + \frac{x^6}{24} + O[x]^8$$

$\mathbf{scos + (x^2 - x - 1)}$

$$-x + \frac{x^2}{2} + \frac{x^4}{24} + O[x]^6$$

$\boldsymbol{D[scos, x]}$

$$-x + \frac{x^3}{6} + O[x]^5$$

Однако вычислить значение выражения или построить график не поучается.

$\mathbf{scos /. x \to 1}$

```
SeriesData::ssdn: Attempt to evaluate a series at the number 1.
Returning Indeterminate.
Indeterminate
```

Чтобы это сделать надо отбросить остаточный член. Функция `Normal` оставляет только степенной многочлен, отбрасывая символ $O\left[x^{n+1}\right]$.

$\mathbf{log1 = Series[Log[1 + x], \{x, 0, 7\}]}$

$$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \frac{x^7}{7} + O[x]^8$$

Normal[**log1**]

$$x-\frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+\frac{x^5}{5}-\frac{x^6}{6}+\frac{x^7}{7}$$

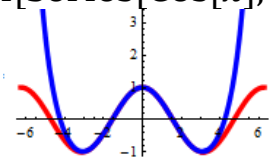
С полученными многочленами можно уже выполнять стандартные операции без всяких ограничений.

Factor[%]

$$\frac{1}{420}x(420-210x+140x^2-105x^3+84x^4-70x^5+60x^6)$$

После отбрасывания «остаточного члена» можно строить график полинома, приближающего исходную функцию. Вот пример графика функции Cos[x] и ее ряда Тейлора до 8-го порядка.

Plot[**Evaluate**[{**Cos**[x], **Normal**[**Series**[**Cos**[x], {x, **0**, **8**}]]}], {x, -2π, 2π}]



Чтобы узнать в каком виде хранится тот или иной объект в системе *Mathematica* можно использовать функцию `InputForm`. Мы уже знаем, что ряды Тейлора хранятся в виде `SeriesData` объекта.

InputForm[**log1**]

SeriesData[x, 0, {1, −1/2, 1/3, −1/4, 1/5, −1/6, 1/7}, 1, 8, 1]

Соответственно, функция `SeriesData[x, x0, {a0, a1, …}, nmin, nmax, den]` создает объект `SeriesData`, который представляет степенной ряд в окрестности точки x_0. Величины a_i являются коэффициентами ряда.

SeriesData[x, **0**, {**1**, $-$**1/2**, **1/3**, $-$**1/4**, **1/5**, $-$**1/6**, **1/7**}, **1**, **8**, **1**]

$$x-\frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+\frac{x^5}{5}-\frac{x^6}{6}+\frac{x^7}{7}+O[x]^8$$

SeriesData[x, **0**, **Table**[$i*\sqrt{i}$, {i, **10**}], **0**, **7**, **1**]

$$1+2\sqrt{2}x+3\sqrt{3}x^2+8x^3+5\sqrt{5}x^4+6\sqrt{6}x^5+7\sqrt{7}x^6+O[x]^7$$

Ряд может быть построен не по целым степеням. Для этого используется последний аргумент `den`, отличный от единицы. В таком случае генерируется ряд с коэффициентами `{a0, a1, …}` при степенях $(x-x_0)^{n_{\min}/den}$, $(x-x_0)^{(n_{\min}+1)/den}$ и т.д. до $(x-x_0)^{n_{\max}/den}$.

SeriesData[x, **0**, {**1**, **1**, **1/2**, **1/6**, **1/24**, **1/120**}, **0**, **6**, **2**]

$$1+\sqrt{x}+\frac{x}{2}+\frac{x^{3/2}}{6}+\frac{x^2}{24}+\frac{x^{5/2}}{120}+O[x]^3$$

Функция `CoefficientList[poly, var]` возвращает список коэффициентов полинома `poly` при степенях переменной `var`.

CoefficientList[(**3** + x)$^{\mathbf{5}}$, x]

```
{243, 405, 270, 90, 15, 1}
```

CoefficientList[(x − **1**)(x − **2**)(x − **3**), x]

```
{-6, 11, -6, 1}
```

CoefficientList[**1** + ***x*** + ***xy*** + ***y*****2** − ***x*****3** + ***y*****3**, {***x***, ***y***}]//**TableForm**

1	0	1	1
1	1	0	0
0	0	0	0
−1	0	0	0

Эта фукция, в каком то смысле, является обратной к `SeriesData`.

se = **Series**[**Exp**[***x***], {***x***, **0**, **5**}]

CoefficientList[**Normal**[**se**], ***x***]

$$\{1,1,\frac{1}{2},\frac{1}{6},\frac{1}{24},\frac{1}{120}\}$$

Функция `SeriesCoefficient[series,n]` возвращает коэффициент при $n-ом$ члене степенного ряда.

SeriesCoefficient[**log1**, **5**]

$$\frac{1}{5}$$

3.5.3 Ряды Фурье

Периодические функции с периодом $2\cdot L$ можно разложить в ряд по синусам и косинусам

$$f(x)=\frac{a_0}{2}+\sum_{n=1}^{\infty}\left[a_n\cos(n\pi x/L)+b_n\sin(n\pi x/L)\right], \tag{1}$$

в котором коэффициенты a_n и b_n вычисляются по формулам

$$\begin{aligned} a_n &= \frac{1}{L}\int_{-L}^{L} f(x)\cos(n\pi x/L)\,dx,\ n=0,1,2,\ldots, \\ b_n &= \frac{1}{L}\int_{-L}^{L} f(x)\sin(n\pi x/L)\,dx,\ n=1,2,\ldots, \end{aligned} \tag{2}$$

В ряд Фурье можно разложить только периодическую функцию. Но если функция определена на конечном интервале, например на интервале $-L\le x\le L,$ то ее всегда можно периодически с периодом $2\cdot L$ продолжить на всю числовую ось и затем использовать разложение для периодической функции. Часто в качестве L выбирают число π. Ряды Фурье удобно записывать в комплексном виде

$$f(x)=\sum_{n=-\infty}^{\infty}c_n e^{i n\pi x/L}, \tag{3}$$

где

$$c_n=\frac{1}{2L}\int_{-L}^{L} f(x)e^{-i n\pi x/L}\,dx,\ n=0,\pm1,\pm2,\ldots. \tag{4}$$

Заметим, что для рядов Фурье имеет место среднеквадратичная сходимость к функции, а не поточечная.

Разложение в ряд Фурье можно интерпретировать как представление функции `f(x)` последовательностью чисел d_n вида

$$d_n=\sqrt{a_n^2+b_n^2}=|c_n| \qquad n=0,1,2,\ldots \tag{5}$$

где коэффициенты d_n служат мерой вклада соответствующих частотных компонент в функцию `f(x)`. Последовательность $\{d_n\}$ называют частотным спектром функции `f`.

В пакете *Mathematica* имеется несколько функций, выполняющих разложение в ряды Фурье.

Функция `FourierTrigSeries[expr,t,n]` раскладывает выражение `expr`, заданное на отрезке $[-\pi, \pi]$ и зависящее от переменной `t`, в тригонометрический ряд Фурье порядка `n`.

FourierTrigSeries$[t^2 - t, t, 3]$

$$\frac{\pi^2}{3} + 4\left(-\text{Cos}[t] + \frac{1}{4}\text{Cos}[2t] - \frac{1}{9}\text{Cos}[3t]\right) + 2\left(-\text{Sin}[t] + \frac{1}{2}\text{Sin}[2t] - \frac{1}{3}\text{Sin}[3t]\right)$$

FourierTrigSeries$[\text{Cos}[t]^2 + \text{Sin}[t]^3, t, 3]$

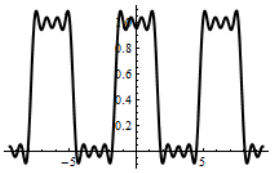
$$\frac{1}{2} + \frac{1}{2}\text{Cos}[2t] + \frac{3\text{Sin}[t]}{4} - \frac{1}{4}\text{Sin}[3t]$$

Точность приближения рядом Фурье разрывных функций хуже, чем непрерывных.

FourierTrigSeries$[\text{UnitBox}[t/\pi], t, 8]$

$$\frac{1}{2} + \frac{2\text{Cos}[t]}{\pi} - \frac{2\text{Cos}[3t]}{3\pi} + \frac{2\text{Cos}[5t]}{5\pi} - \frac{2\text{Cos}[7t]}{7\pi}$$

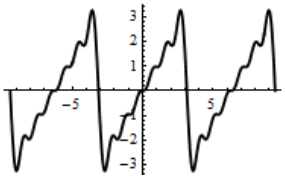
Plot$[\%, \{t, -3\pi, 3\pi\}]$



FourierTrigSeries$[t, t, 6]$

$$2\text{Sin}[t] - \text{Sin}[2t] + \frac{2}{3}\text{Sin}[3t] - \frac{1}{2}\text{Sin}[4t] + \frac{2}{5}\text{Sin}[5t] - \frac{1}{3}\text{Sin}[6t]$$

Plot$[\%, \{t, -3\pi, 3\pi\}]$

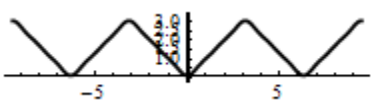


Чем выше гладкость функции, тем лучше приближение при одном и том же порядке ряда/суммы Фурье.

FourierTrigSeries$[\text{Abs}[t], t, 6]$

$$\frac{\pi}{2} - \frac{4\text{Cos}[t]}{\pi} - \frac{4\text{Cos}[3t]}{9\pi} - \frac{4\text{Cos}[5t]}{25\pi}$$

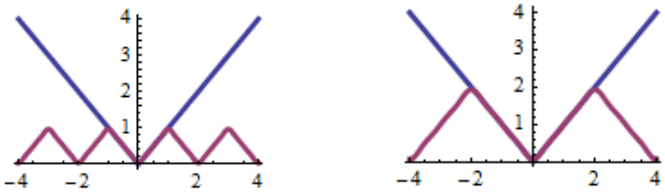
Plot$[\%, \{t, -3\pi, 3\pi\}, \text{AspectRatio} \rightarrow \text{Automatic}]$



При установке опции `FourierParameters->{1,π/L}` создается ряд (1) с коэффициентами (2). Следующий ряд построен на отрезке [-1, 1] и имеет период 2 (следующий рисунок слева).

```
fts1[t_] = FourierTrigSeries[Abs[t], t, 6, FourierParameters → {1, π}]
Plot[{Abs[t], fts1[t]}, {t, −4, 4}, PlotStyle → Thickness[0.01]]
```

$$\frac{1}{2}-\frac{4\text{Cos}[\pi t]}{\pi^2}-\frac{4\text{Cos}[3\pi t]}{9\pi^2}-\frac{4\text{Cos}[5\pi t]}{25\pi^2}$$



Следующий ряд имеет период 4 (L=2).

```
fts2[t_] = FourierTrigSeries[Abs[t], t, 6, FourierParameters → {1, π/2}]
Plot[{Abs[t], fts2[t]}, {t, −4, 4}, PlotStyle → Thickness[0.02]]
```

$$1-\frac{8\text{Cos}[\frac{\pi t}{2}]}{\pi^2}-\frac{8\text{Cos}[\frac{3\pi t}{2}]}{9\pi^2}-\frac{8\text{Cos}[\frac{5\pi t}{2}]}{25\pi^2}$$

(предыдущий рисунок справа).

Опция `Assumptions` уточняет диапазон изменения символьных переменных, используемых в разлагаемой функции.

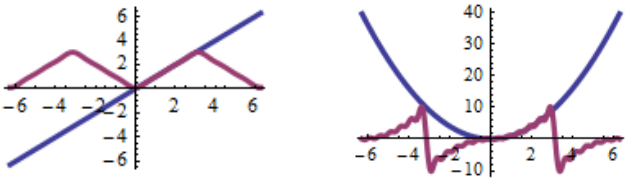
```
FourierTrigSeries[Abs[t − a], t, 3, Assumptions → 0 < a < π]
```

$$\frac{a^2+\pi^2}{2\pi}-\frac{2(1+\text{Cos}[a])\text{Cos}[t]}{\pi}-\frac{2(1+\text{Cos}[3a])\text{Cos}[3t]}{9\pi}+\frac{\text{Cos}[2t]\text{Sin}[a]^2}{\pi}-\frac{2(a+\text{Sin}[a])\text{Sin}[t]}{\pi}+\frac{(a-\text{Cos}[a]\text{Sin}[a])\text{Sin}[2t]}{\pi}-\frac{2(3a+\text{Sin}[3a])\text{Sin}[3t]}{9\pi}$$

В разложении четных функций присутствуют только косинусы. Любую функцию, заданную на отрезке [0,L] можно продолжить четно на отрезок [-L,0], а потом разложить в ряд Фурье. Но можно сразу использовать формулы разложения по косинусам. Функция `FourierCosSeries` выполняет эту работу сразу (следующий рисунок слева).

```
fcs[t_] = FourierCosSeries[t, t, 6]
Plot[{t, fcs[t]}, {t, −2π, 2π}, PlotStyle → Thickness[0.02]]
```

$$\frac{\pi}{2}-\frac{4\text{Cos}[t]}{\pi}-\frac{4\text{Cos}[3t]}{9\pi}-\frac{4\text{Cos}[5t]}{25\pi}$$



Сравните с результатом одного из примеров, приведенных выше, где разлагалась функция Abs[t].

Косинус преобразование функции f(t) эквивалентно преобразованию Фурье функции $\begin{cases} f(t), t \ge 0, \\ f(-t), t < 0 \end{cases}$

В разложении нечетных функций присутствуют только синусы. Любую функцию, заданную на отрезке [0,L], можно нечетно продолжить на отрезок $[-L, 0]$, а потом разложить в ряд Фурье. Но можно сразу использовать

формулы разложения по синусам. Функция `FourierSinSeries` выполняет эту работу сразу.

```
fss[t_] = FourierSinSeries[t^2, t, 12];
Plot[{t^2, fss[t]}, {t, -2π, 2π}, PlotStyle → Thickness[0.02]]
```

(предыдущий рисунок справа).

Синус преобразование функции $f(t)$ эквивалентно преобразованию Фурье функции $\begin{cases} f(t), t \ge 0, \\ -f(-t), t < 0 \end{cases}$.

Функция `FourierSeries` используется для построения комплексных рядов Фурье.

```
fs[t_] = FourierSeries[t, t, 3]
```

$$ie^{-it} - ie^{it} - \frac{1}{2}ie^{-2it} + \frac{1}{2}ie^{2it} + \frac{1}{3}ie^{-3it} - \frac{1}{3}ie^{3it}$$

Легко видеть, что вещественная часть результата совпадает с результатом работы функции `FourierTrigSeries`, а мнимая равна нулю.

```
ComplexExpand[Re[fs[t]]]
```

$$2\mathrm{Sin}[t] - \mathrm{Sin}[2t] + \frac{2}{3}\mathrm{Sin}[3t]$$

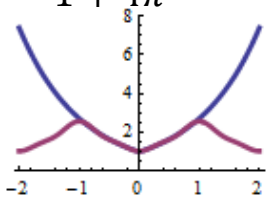
```
ComplexExpand[Im[fs[t]]]
```

0

Опции функций `FourierSeries`, `FourierTrigSeries`, `FourierSinSeries`, `FourierCosSeries`, используются одинаково. Например,

```
fcs[t_] = FourierCosSeries[Exp[t], t, 3, FourierParameters → {1, π}]
Plot[{Exp[Abs[t]], fcs[t]}, {t, -2, 2}, PlotStyle → Thickness[0.01]]
```

$$-1 + e - \frac{2(1+e)\mathrm{Cos}[\pi t]}{1+\pi^2} + \frac{2(-1+e)\mathrm{Cos}[2\pi t]}{1+4\pi^2} - \frac{2(1+e)\mathrm{Cos}[3\pi t]}{1+9\pi^2}$$



Здесь мы построили график четного продолжения функции e^t с положительной полуоси на отрицательную, т.е. график функции $e^{|t|}$; а также косинус разложение функции e^t при L=1 (для этого использовалась опция FourierParameters→{1,π}), которое приближает функцию $e^{|t|}$ на отрезке [-L,L].

Любую функцию этой группы можно использовать для построения ряда Фурье функции нескольких переменных.

```
FourierSeries[x + y, {x, y}, {2, 2}]
```

$$ie^{-ix} - ie^{ix} - \frac{1}{2}ie^{-2ix} + \frac{1}{2}ie^{2ix} + ie^{-iy} - ie^{iy} - \frac{1}{2}ie^{-2iy} + \frac{1}{2}ie^{2iy}$$

```
FourierTrigSeries[x + y, {x, y}, {2, 2}]
```

$$2\mathrm{Sin}[x] - \mathrm{Sin}[2x] + 2\mathrm{Sin}[y] - \mathrm{Sin}[2y]$$

FourierSinSeries[$x * y, \{x, y\}, \{2, 2\}$]

$4\text{Sin}[x]\text{Sin}[y] - 2\text{Sin}[2x]\text{Sin}[y] - 2\text{Sin}[x]\text{Sin}[2y] + \text{Sin}[2x]\text{Sin}[2y]$

FourierCosSeries[$x^2 + y^2, \{x, y\}, \{2, 2\}$]

$\frac{8\pi^2}{3} - 8\text{Cos}[x] + 2\text{Cos}[2x] - 8\text{Cos}[y] + 2\text{Cos}[2y]$

Функция `FourierCoefficient` возвращает n-й коэффициент в разложении функции в ряд Фурье, где n может быть любым целым числом. Так в формате `FourierCoefficient[expr,t,k]` возвращается коэффициент при e^{ikt}. Например,

fc[k_]:= **FourierCoefficient**[t, t, k]

Table[**fc**[k], {$k, -3, 3$}]

$\left\{\frac{i}{3}, -\frac{i}{2}, i, 0, -i, \frac{i}{2}, -\frac{i}{3}\right\}$

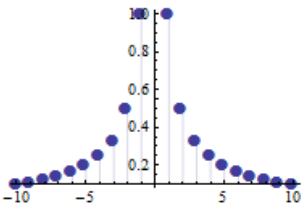
Или

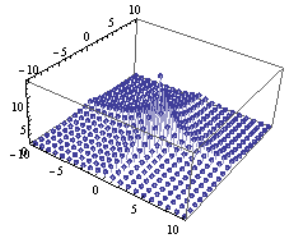
kn[n_] = **FourierCoefficient**[t, t, n]

$\frac{i(-1)^n}{n}$

Имея список коэффициентов или общую формулу для вычисления, можно построить частотный спектр функции (следующий рисунок слева).

DiscretePlot[**Abs**[**kn**[n]], {$n, -10, 10$}, **PlotRange** → **All**,
PlotMarkers → {**Automatic**, **Medium**}]//**Quiet**





fc[m_, n_] = **FourierCoefficient**[**Exp**[$-x - y$], {x, y}, {m, n}]

$-\frac{(-1)^{m+n}\text{Sinh}[\pi]^2}{(-i+m)(-i+n)\pi^2}$

ListPointPlot3D[**Abs**[**Table**[**fc**[m, n], {$m, -10, 10$}, {$n, -10, 10$}]],
Filling → **Bottom**, **PlotRange** → **All**, **DataRange** → {{$-10, 10$}, {$-10, 10$}},
PlotStyle → **PointSize**[**Medium**]] (* предыдущий рисунок справа *)

Аналогично действуют функции `FourierSinCoefficient`, и `FourierCosCoefficient` .

FourierSinCoefficient[$t, t, 3$]

$\frac{2}{3}$

Это коэффициент при $\sin k\,x$. Действительно

Expand[**FourierSinSeries**[$t, t, 4$]]

$2\text{Sin}[t] - \text{Sin}[2t] + \frac{2}{3}\text{Sin}[3t] - \frac{1}{2}\text{Sin}[4t]$

Коэффициенты рядов Фурье вычисляются как интегралы по формулам (2) или (4). Вы можете сравнить.

FourierCoefficient[$t, t, 3$]

$-\frac{i}{3}$

Полученное число это коэффициент при e^{i3t} в разложении Фурье функции $f(t)=t$ на интервале $[-\pi,\pi]$. Тогда, в соответствии с (4), он равен

$\mathbf{1/(2\pi)}$**Integrate**[t **Exp**[$-3It$], {$t, -\pi, \pi$}]

$-\frac{i}{3}$

3.6 Поиск экстремальных значений.

Многие задачи – как из области математики, так и из других областей науки и техники – приводят к задаче о нахождении наибольшего или наименьшего значения некоторой функции. В системе *Mathematica* имеется большой набор функций, предназначенных для решения таких задач.

3.6.1 Экстремумы дискретных наборов

Для поиска минимальных или максимальных значений среди чисел, входящих в список, используются функции `Min` и `Max`.

В формате `Min[x1,x2,...]` или `Max[x1,x2,...]` функция возвращает минимальный/максимальный элемент из набора `x1,x2,...`.

Min[5, 2, 14]

2

Функции работают как с приближенными, так и точными числами.

Min[5, 4. 5, $\sqrt{14}$]

$\sqrt{14}$

В формате `Min[{x1,x2,...},{y1,y2,...},...]` функция возвращает минимальный элемент из всех списков. Аналогичный формат есть у функции `Max`.

Min[{4, 1, 7, 2}]

1

Min[{3, 4, 3}, {2, $\sqrt{2}$}, 7]

$\sqrt{2}$

$M = \{\{-2, 0, 3, 2\}, \{0, 2, 7, 6\}, \{-6, -2, -4, 0\}\}$;

Min[M]

-6

Но эти функции можно использовать и для поиска минимума/максимума каждой строки или каждого столбца матрицы.

Min/@M

```
{-2,0,-6}
```

Min/@Transpose[M]

```
{-6,-2,-4,0}
```

Напомним, что операция /@ является инфиксной формой функции Map, которая в формате Map[f,expr] применяет функцию f к каждому элементу первого уровня в выражении expr. В нашем примере функция Min проносится внутрь внешнего списка и применяется к каждому внутреннему списку матрицы M.

Функции можно использовать с символьными переменными

$$\mathbf{Min}[x, y, \mathbf{Min}[x, z]]$$

```
Min[x,y,z]
```

и выполнять алгебраические преобразования

$$\mathbf{FullSimplify}[\mathbf{Max}[x, y] - \mathbf{Abs}[x - y]]$$

$$\begin{cases} x & x < y \\ y & \text{True} \end{cases}$$

$$\mathbf{FullSimplify}[\mathbf{Max}[x, y] - \mathbf{Abs}[x - y] - \mathbf{Min}[x, y]]$$

0

Функции Min и Max можно дифференцировать

$$\boldsymbol{D}[\mathbf{Min}[x, x^2], x]$$

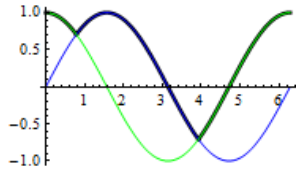
$$\begin{cases} 1 & x < 0 \\ 2x & 0 < x < 1 \\ 1 & x > 1 \\ \text{Indeterminate} & \text{True} \end{cases}$$

Функции Min и Max можно интегрировать.

$$\mathbf{Integrate}[\mathbf{Max}[\mathbf{Sin}[x], \mathbf{Cos}[x]], \{x, \mathbf{0}, \mathbf{2Pi}\}]$$

$$2\sqrt{2}$$

$$\mathbf{Plot}[\{\mathbf{Max}[\mathbf{Sin}[x], \mathbf{Cos}[x]], \mathbf{Sin}[x], \mathbf{Cos}[x]\}, \{x, \mathbf{0}, \mathbf{2Pi}\},$$
$$\mathbf{PlotRange} \to \mathbf{All}, \mathbf{PlotStyle} \to \{\{\mathbf{Black}, \mathbf{Thickness}[\mathbf{0.015}]\}, \mathbf{Blue}, \mathbf{Green}\}]$$



Функции Min и Max можно использовать в граничных значениях итераторов.

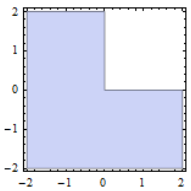
$$\mathbf{Table}[i + j + k, \{i, \mathbf{2}\}, \{j, \mathbf{3}\}, \{k, \mathbf{Max}[i, j], \mathbf{3}\}]$$

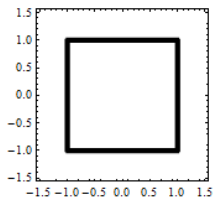
{{{3,4,5},{5,6},{7}},{{5,6},{6,7},{8}}}

Их можно использовать при построении графических объектов.

$$\mathbf{RegionPlot}[\mathbf{Min}[x, y] < 0, \{x, -\mathbf{2}, \mathbf{2}\}, \{y, -\mathbf{2}, \mathbf{2}\}]$$ (*следующий рисунок слева *)

$$\mathbf{ContourPlot}[\mathbf{Max}[\{\mathbf{Abs}[x], \mathbf{Abs}[y]\}] == \mathbf{1}, \{x, -\mathbf{1.5}, \mathbf{1.5}\}, \{y, -\mathbf{1.5}, \mathbf{1.5}\},$$
$$\mathbf{ContourStyle} \to \{\mathbf{Black}, \mathbf{Thickness}[\mathbf{0.03}]\}]$$ (*следующий рисунок справа *)



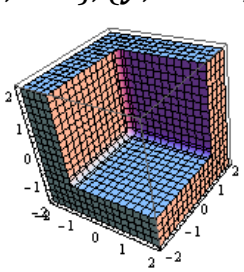
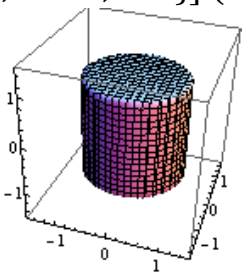
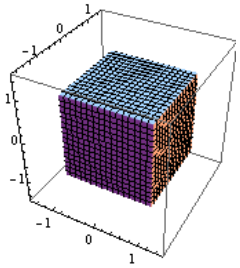


$$\mathbf{RegionPlot3D}[\mathbf{Min}[x, -y, z] < -1,$$
$$\{x, -\mathbf{2}, \mathbf{2}\}, \{y, -\mathbf{2}, \mathbf{2}\}, \{z, -\mathbf{2}, \mathbf{2}\}]$$ (* следующий рисунок слева *)

$$\mathbf{RegionPlot3D}[\mathbf{Max}[x^2 + y^2, z^2] < 1, \{x, -\mathbf{1.5}, \mathbf{1.5}\}, \{y, -\mathbf{1.5}, \mathbf{1.5}\},$$

$\{z, -1.5, 1.5\}$, **PlotPoints** $\rightarrow$ **30**] (* следующий рисунок в центре *)

ContourPlot3D[**Max**[{**Abs**[x], **Abs**[y], **Abs**[z]}] == **1**,
$\{x, -1.5, 1.5\}, \{y, -1.5, 1.5\}, \{z, -1.5, 1.5\}$] (*следующий рисунок справа *)

Близкими по смыслу являются функции `RankedMin` и `RankedMax`. Они могут вычислить n-й наименьший или наибольший элемент списка.

RankedMin[{**1**, **15**, **6**, **21**, **8**, **12**, **13**, **11**}, **3**] (* третий наименьший *)

8

RankedMax[{$\pi, \sqrt{2}, e$}, **2**] (* второй наибольший *)

e

RankedMin[{**1** − x, x, **1** + x}, **1**]//**PiecewiseExpand**

$$\begin{cases} 1-x & x \geq \frac{1}{2} \\ x & \text{True} \end{cases}$$

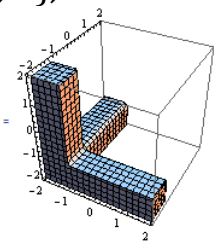
Это то же, что и

Min[{**1** − x, x, **1** + x}, **1**]//**PiecewiseExpand**

Здесь функция `PiecewiseExpand` преобразует вложенные кусочные (piecewise) функции в «одноуровневую» кусочную функцию.

Функции `RankedMin` и `RankedMax` также можно использовать при построении областей.

RegionPlot3D[**RankedMin**[{x, y, z}, **2**] < −1,
$\{x, -2, 2\}, \{y, -2, 2\}, \{z, -2, 2\}$, **PlotPoints** $\rightarrow$ **19**, **Mesh** $\rightarrow$ **Full**]



3.6.2 Экстремумы функций

В системе *Mathematica* для поиска минимума или максимума функции, заданной аналитически, используются функции `Maximize` и `Minimize`. Они имеют одинаковые форматы вызова, поэтому для краткости мы будем описывать каждый формат только для одной из этих функций. Заметим, что эти функции заменили устаревшие функции `ConstrainedMin` и `ConstrainedMax` версии *Mathematica* 5.

В формате `Minimize[expr,x]` определяется минимум выражения `expr` по переменной *x*.

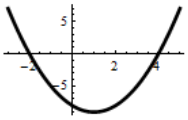
Minimize$[x^2 - 2x - 8, x]$

$\{-9, \{x \to 1\}\}$

Результат возвращается в форме списка `{Мин_значение,{x→x_min}}`.

Вот график нашей функции.

Plot$[x^2 - 2x - 8, \{x, -3, 5\}]$



Поскольку максимум функции $x^2 - 2x - 8$ равен бесконечности, то

Maximize$[x^2 - 2x - 8, x]$

```
Maximize::natt: The maximum is not attained at any point
satisfying the given constraints.≫
```

$\{\infty, \{x \to -\infty\}\}$

и функция `Maximize` вывела сообщение.

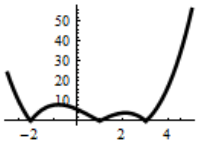
Обратите внимание, что первым аргументом функций `Maximize` и `Minimize` являются выражения, хотя по привычке мы говорим об экстремуме функции.

Исследуемые функции не обязаны быть гладкими.

$f[x\_] = x^3 - 2x^2 - 5x + 6$;

Plot$[$**Abs**$[f[x]], \{x, -3, 5\}]$

Minimize$[$**Abs**$[f[x]], x]$



$\{0, \{x \to -2\}\}$

Функции `Maximize` и `Minimize` умеют работать с символьными выражениями

Minimize$[x^2 - 2ax, x]$

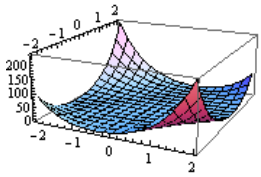
$\{-a^2, \{x \to a\}\}$

Исследуемые на экстремум выражения могут иметь любое конечное количество аргументов.

$f[x\_, y\_] = (x^2 + y^2 + 1)((xy - 1)^2 + 1)$

Plot3D$[f[x, y], \{x, -2, 2\}, \{y, -2, 2\},$ **PlotRange** $\to$ **All**$]$

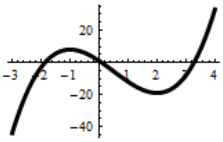
Minimize$[f[x, y], \{x, y\}]$



$\{2, \{x \to 0, y \to 0\}\}$

В формате `Minimize[{expr,ограничения},{x1,x2,...}]` определяется минимум выражения `expr` по переменным `x1,x2,...` с учетом `ограничений` (дополнительных условий).

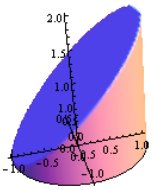
$$f[x\_] = 2x^3 - 3x^2 - 12x + 1;$$
$$\text{Plot}[f[x], \{x, -3, 5\}]$$



$$\text{Minimize}[\{f[x], -2 <= x \leq 3\}, x]$$
$$\{-19, \{x \to 2\}\}$$
$$\text{Maximize}[\{f[x], -2 <= x \leq 3\}, x]$$
$$\{8, \{x \to -1\}\}$$

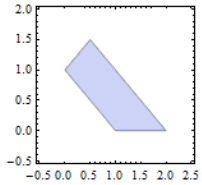
Вот пример для функции 2 – х переменных

$$\text{Minimize}[\{1 + x, x^2 + y^2 \leq 1\}, \{x, y\}]$$
$$\text{Maximize}[\{1 + x, x^2 + y^2 \leq 1\}, \{x, y\}]$$
$$\{0, \{x \to -1, y \to 0\}\}$$
$$\{2, \{x \to 1, y \to 0\}\}$$
$$\text{RegionPlot3D}[x^2 + y^2 \leq 1 \&\& 0 \leq z \leq 1 + x,$$
$$\{x, -1, 1\}, \{y, -1, 1\}, \{z, 0, 2\}, \text{PlotPoints} \to 50, \text{Mesh} \to \text{None},$$
$$\text{Boxed} \to \text{False}, \text{AxesOrigin} \to \{0, 0, 0\}]$$



Вот пример линейных ограничений и линейной целевой функции.

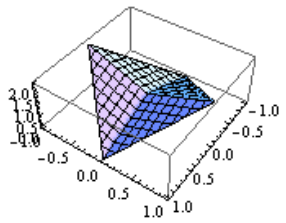
$$\text{Maximize}[\{y, x + y - 1 \geq 0, -x + y - 1 \leq 0, x + y - 2 <= 0, y \geq 0\}, \{x, y\}]$$
$$\{\frac{3}{2}, \{x \to \frac{1}{2}, y \to \frac{3}{2}\}\}$$
$$\text{RegionPlot}[x + y - 1 \geq 0 \;\&\&\; -x + y - 1 \leq 0 \;\&\&\; x + y - 2 \leq 0 \;\&\&$$
$$y \geq 0, \{x, -0.5, 2.5\}, \{y, -0.5, 2\}]$$



Здесь максимум достигается в самой высокой вершине многоугольника, представляющего множество точек, координаты которых удовлетворяют системе неравенств.

Целевая функция, зависящая от нескольких переменных, также не обязана быть гладкой.

$$\text{Minimize}[\{1 - \text{Abs}[x] + \text{Abs}[y], \text{Abs}[x] + \text{Abs}[y] \leq 1\}, \{x, y\}]$$
$$\{0, \{x \to -1, y \to 0\}\}$$
$$\text{Plot3D}[1 - \text{Abs}[x] + \text{Abs}[y], \{x, -1, 1\}, \{y, -1, 1\},$$
$$\text{RegionFunction} \to \text{Function}[\{x, y, z\}, \text{Abs}[x] + \text{Abs}[y] \leq 1]]$$



Целевая функция может быть неограничена. Тогда система может вернуть следующий результат

$\mathbf{Minimize}[\{x+y, xy \le 1\}, \{x, y\}]$

```
Minimize::natt: The minimum is not attained at any point
satisfying the given constraints. ≫
```

$\{-\infty, \{x \to \text{Indeterminate}, y \to \text{Indeterminate}\}\}$

Экстремум может существовать, но не достигаться в области. Тогда результат вычисления может выглядеть следующим образом

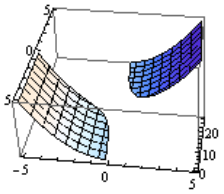
$\mathbf{Minimize}[\{1+x^2, x\,y \ge 1\}, \{x, y\}]$

```
Minimize::natt: The minimum is not attained at any point
satisfying the given constraints. ≫
```

$\{1, \{x \to \text{Indeterminate}, y \to \text{Indeterminate}\}\}$

Следующий график целевой функции проясняет ситуацию.

$\mathbf{Plot3D}[1+x^2, \{x, -5, 5\}, \{y, -5, 5\}, \mathbf{RegionFunction} \to \mathbf{Function}[\{x, y, z\}, xy \ge 1]]$



Глобальный минимум достигается при x=0, а условие $x\,y \ge 1$ исключает значение x=0 из рассмотрения, хотя при достаточно большом y величина x может приблизиться к нулю сколь угодно близко.

Если ограничение задает открытую область (без границы), а экстремум принадлежит границе, то функции `Maximize/Minimize` могут возвратить точку, которая лежит на границе и вывести подходящее сообщение.

$\mathbf{Maximize}[\{xy, x^2+y^2 < 1\}, \{x, y\}]$

```
Maximize::wksol: Warning: there is no maximum in the region in
which the objective function is defined and the constraints are
satisfied; Mathematica will return a result on the boundary. ≫
```

$$\left\{\frac{1}{2}, \left\{x \to -\frac{1}{\sqrt{2}}, y \to -\frac{1}{\sqrt{2}}\right\}\right\}$$

Функции умеют решать задачи целочисленного программирования.

$$\mathbf{Maximize}[\{2x+3y^2-2z^3, 1 \le x+y+z \le 2 \;\&\&\; 1 \le x-y+z \le 2 \;\&\&\; x-y-z == 3 \;\&\&\; \mathbf{Element}[\{x, y, z\}, \mathbf{Integers}]\}, \{x, y, z\}]$$

$\{6, \{x \to 2, y \to 0, z \to -1\}\}$

Здесь к ограничениям – неравенствам мы добавили дополнительное условие `Element[{x,y,z},Integers]`.

Ограничения могут содержать уравнения, неравенства, их логическую комбинацию и, возможно, ограничение «целочисленности». Если выражение `f` и ограничения являются линейными или полиномиальными функции

`Minimize` и `Maximize` будут всегда искать глобальный (в области) экстремум. Если этим функциям переданы выражения, содержащие приближенные числа, то экстремум они будут искать численно (а не символьно), автоматически вызывая функции `NMinimize` и `NMaximize`. Если области «принадлежности» переменных не заданы, то полагается, что все переменные вещественные. Кроме того, следует помнить, что функции возвращают одно значение экстремума, даже если одинаковых экстремальных значений имеется несколько.

Естественно, что функции `Minimize` и `Maximize` успешно решают задачи линейного программирования.

$$\mathbf{Maximize}[\{x-3y-z, 1\le x+y+2z\le 2 \;\&\&\; 1\le x-3y+z\le 2 \;\&\&\; x-2y-5z==3\},\{x,y,z\}]$$

$$\{\frac{12}{5},\{x\to\frac{8}{5},y\to-\frac{1}{5},z\to-\frac{1}{5}\}\}$$

Однако следует знать, что для таких задач в системе имеется специальная функция `LinearProgramming`, которая ожидает входные данные в матричном виде (в виде списков). Например,

$\boldsymbol{c}=\{1,-3,-1\}$; (*коэффициенты целевой функции*)

$\boldsymbol{m}=\{\{1,1,2\},\{1,1,2\},\{1,-3,1\},\{1,-3,1\},\{1,-2,-5\}\}$;(*коэф. ограничений*)

$\boldsymbol{b}=\{\{1,1\},\{2,-1\},\{1,1\},\{2,-1\},\{3,0\}\}$; (*знач. прав. части, тип неравенства*)

$$\mathbf{LinearProgramming}[-\boldsymbol{c},\boldsymbol{m},\boldsymbol{b},-\mathbf{Infinity}]$$

$$\{\frac{8}{5},-\frac{1}{5},-\frac{1}{5}\}$$

Само значение целевой функции находим следующим образом

$$\boldsymbol{c}.\%$$

$$\frac{12}{5}$$

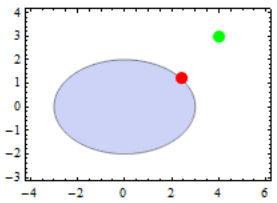
Подробнее с функцией `LinearProgramming` вы можете познакомиться по справочной системе.

Пример. Найдем минимальное расстояние от точки (4,3) до эллипса $\frac{x^2}{9}+\frac{y^2}{4}=1$. Имеем

$$\mathbf{m}=\mathbf{Minimize}[\{(\mathbf{x}-\mathbf{4})^2+(\mathbf{y}-\mathbf{3})^2,\frac{\mathbf{x^2}}{\mathbf{9}}+\frac{\mathbf{y^2}}{\mathbf{4}}\le\mathbf{1}\},\{\mathbf{x},\mathbf{y}\}]$$

$$\{\frac{29}{5},\{x\to\frac{12}{5},y\to\frac{6}{5}\}\}$$

$$\mathbf{RegionPlot}[\frac{\boldsymbol{x^2}}{\mathbf{9}}+\frac{\boldsymbol{y^2}}{\mathbf{4}}\le\mathbf{1},\{\boldsymbol{x},-\mathbf{4},\mathbf{6}\},\{\boldsymbol{y},-\mathbf{3},\mathbf{4}\},$$
$$\mathbf{Epilog}\to\{\mathbf{PointSize}[\mathbf{0.05}],\{\mathbf{Red},\mathbf{Point}[\{\boldsymbol{x},\boldsymbol{y}\}/.\boldsymbol{m}[[\mathbf{2}]]]\},$$
$$\{\mathbf{Green},\mathbf{Point}[\{\mathbf{4},\mathbf{3}\}]\}\},\mathbf{AspectRatio}\to\mathbf{Automatic}]$$



□

Пример. Среди всех прямоугольников единичного периметра найти прямоугольник с максимальной площадью. Обозначив через x и y длины сторон прямоугольника, получаем

$$\mathbf{Maximize}[\{x\,y,\ 2x+2y==1\ \&\&\ x>0\ \&\&\ y>0\},\{x,y\}]$$

$$\{\frac{1}{16},\{x\to\frac{1}{4},y\to\frac{1}{4}\}\}$$

□

Пример. Среди всех прямоугольных треугольников единичной площади найти треугольник с минимальным периметром. Обозначив через x и y длины сторон прямоугольника, имеем $S=\frac{1}{2}x\,y$, $P=x+y+\sqrt{x^2+y^2}$. Тогда

$$\mathbf{Minimize}[\{x+y+\sqrt{x^2+y^2},x\,y==2\ \&\&\ x>0\ \&\&\ y>0\},\{x,y\}]$$

$$\{2(1+\sqrt{2}),\{x\to\cdots,y\to\cdots\}\}$$

$$N[\%]$$

```
{4.8284271,{x→1.4142136,y→1.4142136}}
```

□

Пример. Найти расстояние от заданной точки до параболы.

$$\mathbf{x0=3;y0=3;Minimize}[\{\mathbf{Sqrt}[(x-\mathbf{x0})^2+(y-\mathbf{y0})^2],y==x^2\},\{x,y\}]$$

$$\{\frac{1}{2}\sqrt{43-14\sqrt{7}},\{x\to\frac{1}{2}(1+\sqrt{7}),y\to\frac{1}{4}(1+\sqrt{7})^2\}\}$$

$$\mathbf{x0=3;y0=6;Minimize}[\{\mathbf{Sqrt}[(x-\mathbf{x0})^2+(y-\mathbf{y0})^2],y==x^2\},\{x,y\}]$$

$$\{\sqrt{\text{Root}[-6057+21157\#1-1192\#1^2+16\#1^3\&,1]},$$
$$\{x\to\text{Root}[-3-11\#1+2\#1^3\&,3],y\to\text{Root}[-3-11\#1+2\#1^3\&,3]^2\}\}$$

$$N[\%]$$

```
{0.53948303,{x→2.4712314,y→6.1069848}}
```

Вы можете аналогично находить расстояние от точки до любой кривой, заменив уравнение кривой.

□

Функции `MinValue`, `ArgMin`, `MaxValue` и `ArgMax` решают те же задачи, которые решают функции `Maximize` и `Minimize`. Функции `MinValue` и `MaxValue` возвращают значение минимума и максимума, а функции `ArgMin`, и `ArgMax` возвращают координаты точки, в которой экстремум достигается. Фактически функция `MinValue[...]` эквивалентна команде `First[Minimize[...]]`, а функция `ArgMin[...]` эквивалентна команде `{x,y,...}/.Last[Minimize[...,{x,y,...},...]`. Аналогичное замечание касается функций `MaxValue` и `ArgMax`.

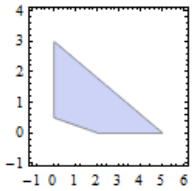
$$\mathbf{MinValue}[\{(x+y-1)^2+1, x^2+y^2 \le 1\}, \{x, y\}]$$
1
$$\mathbf{ArgMin}[\{(x+y-1)^2+1, x^2+y^2 \le 1\}, \{x, y\}]$$
$$\{\frac{1}{2}, \frac{1}{2}\}$$
$$\mathbf{Minimize}[\{(x+y-1)^2+1, x^2+y^2 \le 1\}, \{x, y\}]$$
$$\{1, \{x \to \frac{1}{2}, y \to \frac{1}{2}\}\}$$
Функции `NMaximize`, `NMinimize` численно находят максимум/минимум функции/выражения. Они возвращают результат в такой же форме как и функции `Maximize` и `Minimize`. Их также можно применять для решения задач целочисленного программирования.
Для линейной целевой функции и линейных ограничений эти функции определяют глобальный экстремум. Для других типов выражений только гарантируется, что будет определен локальный экстремум, хотя функции все же пытаются найти глобальное значение.

В формате `NMinimize[expr,x]` численно определяется минимум выражения `expr` по переменной *x*. Аналогичный формат имеется у функции `NMaximize`.

В формате `NMinimize[expr,{x1,x2,...}]` численно определяется минимум выражения `expr` по переменным $x_1, x_2, \ldots$. Аналогичный формат имеется у функции `NMaximize`.

В формате `NMinimize[{expr,ограничения},{x1,x2,...}]` численно определяется минимум выражения `expr` по переменным $x_1, x_2, \ldots$ с учетом `ограничений` (дополнительных условий). То же имеет место для функции `NMaximize`.
$$\mathbf{NMaximize}[-x^6 - 3x^4 + x, x]$$
```
{0.32123215,{x→0.42441332}}
```
$$\mathbf{NMaximize}[\{x^4 - (y-1)^4, x^2+y^2 \le 2\}, \{x, y\}]$$
```
{3.4397099,{x→1.3949862,y→0.23240812}}
```
$$\mathbf{NMaximize}[\{\mathbf{Exp}[\mathbf{Sin}[x]] - \mathbf{Sin}[10(x+y)] + (x^2+y^2)/4, x^2+y^2 \le 1\}, \{x, y\}]$$
```
{3.442848,{x→0.90372233,y→-0.42811908}}
```
В системе ограничений допустимо использовать операцию || (или).
$$\mathbf{NMaximize}[\{x+y, x^2+y^2 \le 1 || (x-1)^2+y^2 \le 1\}, \{x, y\}]$$
```
{2.4142136,{x→1.7071065,y→0.70710707}}
```
Можно потребовать, чтобы некоторые из искомых переменных были целыми числами.
$$\mathbf{NMaximize}[\{-x-3y, 3x+5y \le 15 \;\&\&\; x+4y \ge 2 \;\&\&\; x \ge 0 \;\&\&\; y \ge 0 \;\&\&\; \{x, y\} \in \mathbf{Integers}\}, \{x, y\}]$$
$$\{-2., \{x \to 2, y \to 0\}\}$$
$$\mathbf{NMaximize}[\{-x-3y, 3x+5y \le 15 \;\&\&\; x+4y \ge 2 \;\&\&\; x \ge 0 \;\&\&\; y \ge 0 \;\&\&\; \{x\} \in \mathbf{Integers}\}, \{x, y\}]$$
```
{-1.5,{x→0,y→0.5}}
```

RegionPlot[$3x+5y\le 15$ && $x+4y\ge 2$ && $x\ge 0$ && $y\ge 0$,
$\{x,-1,6\},\{y,-1,4\}$]



В некоторых случаях функции `NMaximize` и `NMinimize` могут заменять функции `Solve` и `NSolve`. В следующем примере функция `NMaximize` находит единственный вещественный корень уравнения

NMaximize[$\{1, x^3+x+1==0\},\{x\}$]

```
{1.,{x→-0.68232781}}
```

Сравните

NSolve[$x^3+x+1==0, x$]

$\{\{x\to -0.68232781\},$
$\{x\to 0.3411639-1.1615414i\},$
$\{x\to 0.3411639+1.1615414i\}\}$

Для функций `NMaximize` и `NMinimize` характерно использование опций, которые уместно использовать в численных алгоритмах.

Опция `WorkingPrecision` задает количество значащих цифр, используемых во внутренних вычислениях; установка `WorkingPrecision->MachinePrecision` (по умолчанию) приводит к вычислениям с процессорной точностью.

Опция `AccuracyGoal->n` задает количество значащих цифр абсолютной погрешности, которая считается равной 10^{-n}. Опция `PrecisionGoal->m` определяет «относительную погрешность» $|x|\cdot 10^{-m}$ вычислений и задается количеством значащих цифр. Установкой по умолчанию для этих опций является `WorkingPrecision/2`. Например, команда

NMinimize[**Sin**[x], x, **AccuracyGoal** → **12**, **PrecisionGoal** → **8**]

```
{-1.,{x→-1.5707963}}
```

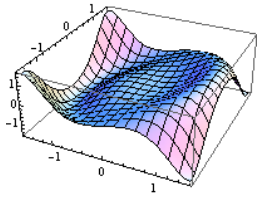
определяет следующий критерий остановки вычислений (погрешность) $\left\| x-x^* \right\| \le \max\left(10^{-12}, 10^{-8}\,x\right)$ и $\nabla f(x)\le 10^{-12}$. Первое неравенство относится к координатам точки экстремума, второе – к значению целевой функции.

Опцией `EvaluationMonitor` задается набор команд, которые вычисляются каждый раз, когда вычисляется значение выражения, стоящего в функции `NMinimize`.

Пример. Найдем максимальное значение функции $f(x,y)=x\sin\left(x^2+y^2\right)$ в области $-\pi/2\le x\le \pi/2$, $-\pi/2\le y\le \pi/2$. Вначале построим график функции.

f[x_, y_]:= x**Sin**[x^2+y^2]

Plot3D[$f[x,y]$, $\{x,-\pi/2,\pi/2\}$, $\{y,-\pi/2,\pi/2\}$, **PlotRange** → **All**]



Мы видим, что есть экстремумы в угловых точках области и локальные экстремумы внутри. Найдем их.

$$\mathbf{Maximize}[\{f[x,y], -\pi/2 \le x <= \pi/2, -\pi/2 \le y <= \pi/2\}, \{x,y\}]$$

$$\{-\frac{1}{2}\pi \mathrm{Sin}[\frac{1}{4}(6-\pi)\pi + \frac{\pi^2}{4}], \{x \to -\frac{\pi}{2}, y \to -\frac{1}{2}\sqrt{(6-\pi)\pi}\}\}$$

$$N[\%]$$

```
{1.5707963,{x→-1.5707963,y→-1.4983284}}
```

Функция `Maximize` нашла максимум в области (один из двух имеющихся). Теперь посмотрим, что даст функция `NMaximize`.

$$\mathbf{NMaximize}[\{f[x,y], -\pi/2 \le x <= \pi/2, -\pi/2 \le y <= \pi/2\}, \{x,y\}]$$

```
{1.3076194,{x→1.3552111,y→3.010915·10^-9}}
```

Функция `NMaximize` нашла локальный максимум внутри области.

Изобразим последовательность «пробных» точек, которые использовала функция `NMaximize` при приближении к локальному максимуму. Вначале запомним координаты «пробных» точек, которые расположены недалеко от экстремума. Для этого можно использовать опцию `EvaluationMonitor`.

$$\{\mathbf{sol}, \mathbf{pts}\} = \mathbf{Reap}[\mathbf{NMaximize}[\{f[x,y], -\pi/2 \le x \le \pi/2, -\pi/2 \le y \le \pi/2\},$$
$$\{x,y\}, \mathbf{EvaluationMonitor} :\to \mathbf{Sow}[\{x,y\}]]];$$

Напомним, что значение опции `EvaluationMonitor` является выражение, которое вычисляется каждый раз, когда вычисляется значение целевой функции. Функция `Sow` «выбрасывает в окружающее пространство» результат вычисления своего аргумента (в нашем случае пару {*x*,*y*}, содержащую координаты точки). Эти пары из «окружающего пространства подбирает» ближайшая объемлющая функция `Reap`. Первым элементом она возвращает результат вычисления основного выражения (в нашем случае результат помещается в переменную **sol**). Вторым элементом, который она возвращает, является список «собранных» ею пар {*x*,*y*} координат «пробных» точек. Можете посмотреть, чем являются переменные **sol** и **pts**.

$$\mathbf{sol}$$

```
{1.3076194,{x→1.3552111,y→3.010915·10^-9}}
```

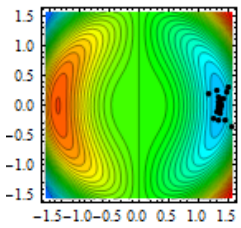
$$\mathbf{pts}$$

```
{{{1.4894175,1.0840092},{-1.0709764,-0.44463881},...}
```

Мы не стали здесь приводить длинный список координат «пробных» точек.

Теперь, имея координаты «пробных» точек, выделим и изобразим только те, которые находятся рядом с экстремумом.

$$\mathbf{ContourPlot}[f[x,y], \{x, -\pi/2, \pi/2\}, \{y, -\pi/2, \pi/2\},$$
$$\mathbf{ColorFunction} \to (\mathbf{Hue}[0.6\#]\&), \mathbf{Contours} \to 30,$$
$$\mathbf{Epilog} \to \{\mathbf{Black}, \mathbf{PointSize}[0.03], \mathbf{Map}[\mathbf{Point},$$
$$\mathbf{Cases}[\mathbf{First}[\mathbf{pts}], z\_/; \mathbf{Abs}[f@@z - \mathbf{First}[\mathbf{sol}]] \le 0.5]]\}]$$



Здесь выражение First[pts] имееет вид

First[pts]

```
{{1.4894175,1.0840092},{-1.0709764,-0.44463881},...}
```

Функция `Cases[{e1,e2,...},шаблон]` возвращает список элементов e_i, которые соответствуют шаблону. В нашем случае элементами являются пары чисел $\{x_i,y_i\}$. В результате шаблон z_ представляет список таких пар (т.е. он имеет вид List[x_i,y_i]). Условная операция /; (слэш – точка с запятой) выделяет из всех пар $\{x_i,y_i\}$ только те, которые удовлетворяют условию, записанному после операции. В нашем случае выделяются только те точки/списки $\{x_i,y_i\}$, в которых значение целевой функции f[x_i,y_i] находится близко к найденному экстремальному значению First[sol]. Само значение f[x_i,y_i] получается использованием операции @@ (функции Apply), которая в нашем случае действует следующим образом: f@@z=Apply[f,{xi,yi}]= f[xi,yi].

Напомним, что функция `Apply[f,expr]` заменяет именем `f` заголовок выражения `expr` (в нашем случае `List`). С другими функциями, использованными в коде, мы знакомились ранее.

□

В функциях `NMaximize` и `NMinimize` можно использовать несколько разных алгоритмов численного поиска экстремума, которые можно задать с помощью опции `Method`. Иногда выбор одного из них, отличного от выбора по умолчанию, может дать лучший результат. Мы не ставим перед собой задачу изложения этих методов. С ними вы можете познакомиться по справочной системе.

У функций `NMaximize` и `NMinimize` есть опция `MaxIterations` , само название которой говорит о ее назначении, и опция `StepMonitor`, с которой мы познакомимся позже.

Для поиска локального минимума/максимума функции $f(x)$, в окрестности значения $x = x_0$, используются функции `FindMinimum` и `FindMaximum`. Они имеют идентичные форматы вызова, поэтому для краткости мы будем описывать форматы только одной из них. Результат возвращается в виде таком же, в каком возвращают результат функции `NMaximize` и `NMinimize`.

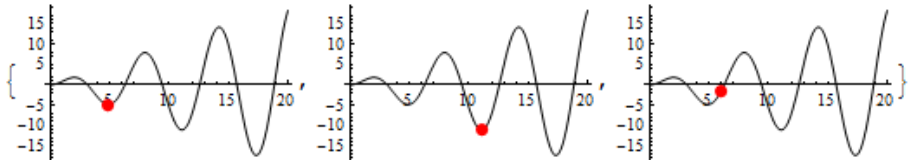
В формате `FindMinimum[f[x],{x,x0}]` численно ищется локальный минимум выражения `f[x]`; начальное значение задается точкой x_0. В формате `FindMinimum[f[x],x]` численно ищется локальный минимум выражения `f[x]`, начиная поиск из случайно выбираемой точки.

В формате `FindMinimum[f[x,y,...],{{x,x0},{y,y0},...}]` численно ищется локальный минимум функции нескольких переменных в окрестности точки `{x0,y0,...}`. В формате `FindMinimum[{f[x,y,...],ограничения},{{x,x0},{y,y0},...}]` численно ищется локальный минимум с учетом `ограничений`. При этом точку `{x0,y0,...}` можно не задавать – она будет автоматически выбираться в области, определяемой ограничениями.

В следующем коде мы вызываем функцию `FindMinimum` три раза с разными начальными приближениями. При этом в третьем вызове мы задаем еще интервал изменения независимой переменной. Положение минимума, каждый раз разное, показано на графиках.

```
f[x_] = x Sin[x];
s1 = FindMinimum[f[x], {x, 3}]
s2 = FindMinimum[f[x], {x, 10}]
s3 = FindMinimum[{f[x], 6 ≤ x ≤ 15}, {x, 7}]
gplot[ss_] := Plot[f[x], {x, 0, 20}, Epilog → {xr = x /. ss[[2]];
                                {Red, PointSize[0.05], Point[{xr, f[xr]}]}}];
{gplot[s1], gplot[s2], gplot[s3]}
```

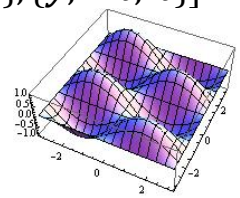
```
{-4.8144699,{x→4.9131804}}
{-11.040708,{x→11.085538}}
{-1.6764929,{x→6.}}
```



Как видим, выбор начальной точки влияет на положение точки минимума. В третьем вызове ближайший к начальной точке минимум оказался на границе области.

Аналогично для функции многих переменных, стартуя из разных точек, получаем разные локальные экстремумы. Рассмотрим функцию график которой показан на следующем рисунке.

```
Plot3D[Sin[x]Sin[2y], {x, -π, π}, {y, -π, π}]
```



Вызовем функцию `FindMinimum` с разными начальными точками.

```
FindMinimum[Sin[x]Sin[2y], {{x, 2}, {y, 2}}]
```

```
{-1.,{x→1.5707963,y→2.3561945}}
```

```
FindMinimum[Sin[x]Sin[2y], {{x, 0}, {y, 1}}]
```

```
{-1.,{x→-1.5707963,y→0.78539816}}
```

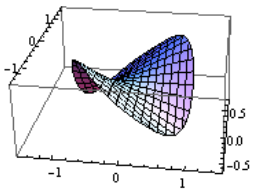
При задании области ограничений переменных начальную точку задавать не обязательно

FindMinimum[{**Sin**[x]**Sin**[$2y$], $x^2 + y^2 \le 1$}, {x, y}]

```
{-0.66871014,{x→-0.79512351,y→0.60644782}}
```

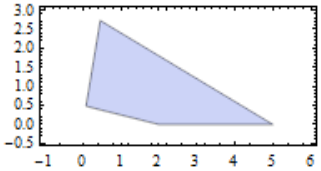
На следующем рисунке показана целевая функция в области единичного круга, который использовался при поиске экстремума.

Plot3D[**Sin**[x]**Sin**[$2y$], {$x, -1.5, 1.5$}, {$y, -1.5, 1.5$},
RegionFunction → **Function**[{x, y, z}, $x^2 + y^2 \le 1$]]



Ограничения могут содержать условие «целочисленности» некоторых (или всех) переменных. Зададим следующую область

RegionPlot[$3x + 5y \le 15$ && $x + 4y \ge 2$ && $y \le 6x$ && $y \ge 0$,
{$x, -1, 6$}, {$y, -0.5, 3$}, **AspectRatio** → **Automatic**]



Теперь найдем минимум функции x+y в этой области, задавая условие «целочисленности» переменной y, и не задавая его.

FindMinimum[{$x + y, 3x + 2y \le 7$ && $x + 2y \ge 6$ && $y \le 10x$ &&
$y \ge 0$ && $y \in$ **Integers**}, {x, y}]

```
{3.3,{x→0.3,y→3}}
```

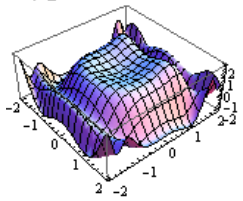
FindMinimum[{$x + y, 3x + 2y \le 7$ && $x + 2y \ge 6$ && $y \le 10\,x$ &&
$y \ge 0$}, {x, y}]

```
{3.1428571,{x→0.28571429,y→2.8571429}}
```

Естественно, что результат оказался разный.

Для функций `FindMinimum` и `FindMaximum` характерно использование опций, которые уместно использовать в численных алгоритмах. Опции `WorkingPrecision`, `AccuracyGoal`, `PrecisionGoal` используются точно также, как было описано для функций `NMaximize` и `NMinimize`. Опция `Method` имеет другие значения, поскольку алгоритмы поиска локального экстремума отличаются от алгоритмов поиска глобального. Новым является добавление опции `Gradient`. Она задает выражение для вычисления градиента минимизируемой / максимизируемой функции.

f[x_, y_]:= **Cos**[$x^2 - y^3$] + **Sin**[$x^2 + y^2$];
Plot3D[f[x, y], {$x, -2, 2$}, {$y, -2, 2$}]



```
FindMinimum[f[x, y], {x, y}, Gradient :>
    {2xCos[x^2 + y^2] - 2xSin[x^2 - y^3], 2yCos[x^2 + y^2] + 3y^2Sin[x^2 - y^3]}]
{-2., {x→1.3446341, y→1.7042148}}
```

Опция `StepMonitor` позволяет контролировать шаги вычислений.

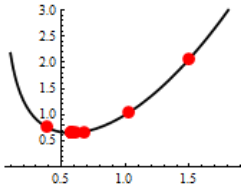
```
f[x_] = x^2 + 1/√x - 1;
FindMinimum[f[x], {x, 0.7}, StepMonitor :> Print["x = ", x]]
x = 0.57904634
x = 0.573647
x = 0.57435347
x = 0.57434918
{0.64938489, {x→0.57434918}}
```

Можно, например, контролировать количество итераций

```
c = 0; {FindMinimum[f[x], {x, 0.7}, StepMonitor :> c++], c}
{{0.64938489, {x→0.57434918}}, 4}
```

Используя эту опцию, можно изобразить промежуточные точки на каждом шаге вычислений.

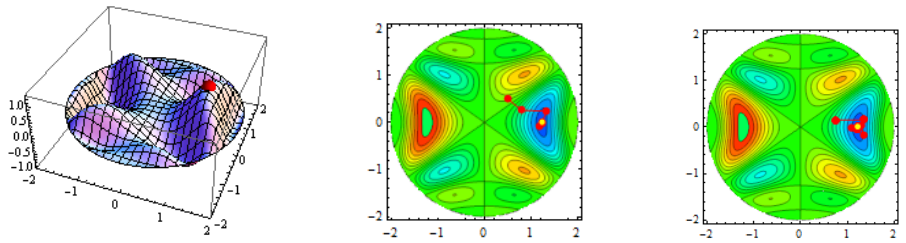
```
{sol, pts} = Reap[FindMinimum[f[x], {x, 3},
                    StepMonitor :> Sow[{x, f[x]}]]]
Plot[f[x], {x, 0.1, 2}, Epilog → {Red, PointSize[0.05],
                    Map[Point, pts[[1]]]}, PlotRange → {0, 3}]
```



У опций `EvaluationMonitor` и `StepMonitor` есть различие в моменте вызова. Первая вызывается после каждого вычисления значения входной функции, вторая – после каждого шага итераций.

Пример. Изобразим путь, по которому двигаются точки алгоритма, приближая максимум функции $f(x,y)=\sin\left(x^2-y^2\right)\left(4-x^2-y^2\right)/3$ в области $x^2+y^2\le 4$. На следующем рисунке слева показана график функции в области ограничений и точка экстремума, который нашла функция `FindMinimum`.

```
f[x_, y_] := x Sin[x^2 - 2y^2](4 - x^2 - y^2)/3;
ps = Plot3D[f[x, y], {x, -2, 2}, {y, -2, 2},
    PlotRange → {{-2, 2}, {-2, 2}, {-1, 1}}, BoxRatios → {4, 4, 2},
    RegionFunction → Function[{x, y, z}, x^2 + y^2 ≤ 4], Mesh → 41];
{sol, pts} = Reap[FindMaximum[{f[x, y], x^2 + y^2 ≤ 4},
        {{x, 0.5}, {y, 0.5}}, StepMonitor :> Sow[{x, y}]]];
{xp, yp} = {x, y} /. sol[[2]]; zp = sol[[1]];
pm = Graphics3D[{Red, Sphere[{xp, yp, zp}, 0.15]}];
Show[ps, pm, PlotRange → {{-2, 2}, {-2, 2}, {-1, 1.1}}] (* рисунок слева *)
```



На рисунке в середине показан контурный график исследуемой функции и ломаная, вершины которой представляют точки последовательного приближения. Желтая точка показывает положение максимума, который определила функция `FindMaximum`.

```
ContourPlot[f[x, y], {x, -2, 2}, {y, -2, 2}, ColorFunction → (Hue[0.6#]&),
  Contours → 21, RegionFunction → Function[{x, y, z}, x^2 + y^2 ≤ 4],
  Epilog → {Red, Line[pts], PointSize[0.04], Point[pts], Yellow,
    PointSize[0.03], Point[{xp, yp}]}] (* пред. рисунок в центре *)
```

Для сравнения изобразим ломаную, по которой функция `NMaximize` приближается к максимуму.

```
{sol2, pts2} = Reap[NMaximize[{f[x, y], x^2 + y^2 ≤ 4}, {x, y},
                              StepMonitor :→ Sow[{x, y}]]];
{xp2, yp2} = {x, y}/. sol2[[2]];

ContourPlot[f[x, y], {x, -2, 2}, {y, -2, 2},
  ColorFunction → (Hue[0.6#]&), Contours → 21,
  RegionFunction → Function[{x, y, z}, x^2 + y^2 ≤ 4],
  Epilog → {Red, Line[pts2], PointSize[0.04], Point[pts2],
    Yellow, PointSize[0.03], Point[{xp2, yp2}]}]
```

Полученный контурный график представлен на предыдущем рисунке справа. Напомним, что начальное приближение в области ограничений (если оно не задано) функция `NMaximize` выбирает самостоятельно.

□

3.7 Приближение функций.

Когда функция задана в дискретном наборе точек и требуется определить ее значение в промежуточных точках, то прибегают к интерполяции или аппроксимации. При этом неизвестная истинная функция заменяется некоторой другой, часто кусочной, имеющей достаточно простые выражения на кусках. Если приближающая функция принимает заданные значения в узлах, то говорят об интерполяции, если нет, то говорят об аппроксимации.

3.7.1 Полиномиальная интерполяция

Функция `InterpolatingPolynomial` создает полином (выражение, степенной многочлен), который проходит через узловые точки. В одномерном случае по `n` точкам строится полином степени `n-1`.

При вызове функции `InterpolatingPolynomial` ей передается список координат узлов и идентификатор переменной полинома, например, *x*, поскольку возвращаться будет выражение, а не функция. При этом интерполяционный полином возвращается в форме Горнера. Например,

InterpolatingPolynomial[{**1**, **8**, **27**, **64**, **125**}, ***x***]

$1+(-1+x)(7+(-2+x)(3+x))$

Здесь список `{1,8,27,64,125}` представляет ординаты узловых точек; их абсциссами полагаются натуральные числа `{1,2,3,...}`. Результирующий полином всегда можно преобразовать к обычному виду.

Expand[%]

x^3

Существует бесчисленное множество полиномов, проходящих через заданные точки. Функция `InterpolatingPolynomial` пытается возвратить полином наименьшей степени.

В формате `InterpolatingPolynomial[{{x₁,y₁},{x₂,y₂},...},x]` задаются обе координаты узловых точек.

В следующем коде мы выполняем полиномиальную интерполяцию по множеству точек, приближенно расположенных на параболе $y=x^2$; смещение с параболы задается небольшим случайным значением.

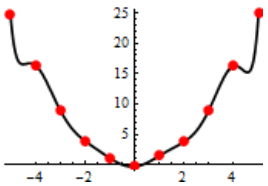
SeedRandom[ ];

data = **Table**[{***i***, ***i***$^{\mathbf{2}}$ + **RandomReal**[{−**0.5**, **0.5**}]}, {***i***, −**5**, **5**}]

g[***x***_] = **Collect**[**InterpolatingPolynomial**[**data**, ***x***], ***x***]

Plot[***g***[***x***], {***x***, −**5**, **5**}, **PlotStyle** → {**Black**, **Thickness**[**0.01**]},
Epilog−> {**Red**, **PointSize**[**0.04**], **Point**/@**data**}]

`-0.29820805+0.33587531 x+1.9489638 x^2-0.18505705 x^3-0.42240753 x^4+0.029138328 x^5+0.061215827 x^6-0.0017099349 x^7-0.0034036856 x^8+0.000032814108 x^9+0.000062829842 x^10`



Здесь функция `SeedRandom[ ]` инициализирует генератор случайных чисел текущим временем.

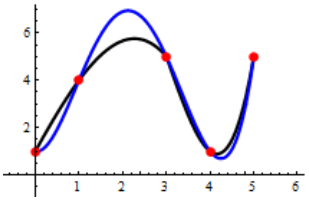
В следующем пункте мы рассмотрим кусочно – полиномиальную интерполяцию. Забегая вперед, выполним сравнение кусочно – полиномиальной интерполяции, выполняемой функцией `Interpolation`, и полиномиальной, выполняемую функцией `InterpolatingPolynomial`.

data = {{**0**, **1**}, {**1**, **4**}, {**3**, **5**}, {**4**, **1**}, {**5**, **5**}};

f = **Interpolation**[**data**];

```
g[x_] = Collect[InterpolatingPolynomial[data, x], x]
Show[Plot[{f[x], g[x]}, {x, 0, 5}, PlotStyle → {Black, Blue}],
     ListPlot[data, PlotStyle → {Red, PointSize[0.03]}],
     PlotRange → {{−1, 6}, {−1, 7}}, AxesOrigin → {0, 0}]
```

$$1-\frac{11x}{30}+\frac{683x^2}{120}-\frac{79x^3}{30}+\frac{37x^4}{120}$$



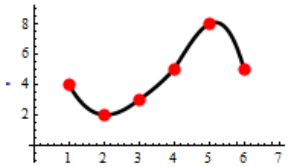
3.7.2 Кусочно – полиномиальная интерполяция

Для представления результатов интерполяции в *Mathematica* используется специальный объект `InterpolatingFunction`. Это «чистая» функция, которую можно использовать как обыкновенную функцию: вычислять значение в точке, дифференцировать и т.д. Единственное ограничение состоит в том, что она создается на определенном диапазоне изменения аргументов. При попытке подстановки аргумента, выходящего за границы этого диапазона, генерируется сообщение.

`InterpolatingFunction` выполняет кусочно – полиномиальную интерполяцию, порядок гладкости которой можно задать при создании функции. Она обычно создается другими функциями системы, выполняющими приближенные вычисления. В частности при численном решении ДУ с помощью функции `NDSolve` создается приближенное решение в виде такой функции.

Алгебраическую интерполяцию в системе выполняет функция `Interpolation`, которая принимает набор точек и создает объект `InterpolatingFunction`.

```
data = {4, 2, 3, 5, 8, 5};
f = Interpolation[data]
f[2.5]
Show[Plot[f[x], {x, 1, 6}, PlotStyle → {Black, Thickness[0.015]}],
     ListPlot[data, PlotStyle → {Red, PointSize[0.05]}],
     PlotRange → {{0, 7}, {−1, 9}}, AxesOrigin → {0, 0}]
InterpolatingFunction[{{1,6}}," <> "]
2.25
```



Здесь мы использовали формат вызова `Interpolation[{y1,y2,...}]`, который полагает, что абсциссы точек равны `{1,2,...}`, а ординаты – `{y1,y2,...}`.

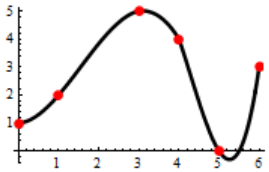
При выводе функции `InterpolatingFunction[{{1,6}},"<>"]` показан диапазон интерполяции `{1,6}`. Именно его мы использовали в качестве диапазона при построении графика функции `f[x]`. Попробуйте использовать более широкий диапазон `Plot[f[x],{x,0,7},...` и вы получите предупреждающее сообщение, хотя график будет построен.

Если нужно вычислить значение интерполяционной функции в точке, то можно использовать следующий формат `Interpolation[data,`$x_{точки}$`]`.

```
Interpolation[data, 2.5]
2.25
```

В формате вызова `Interpolation[{{x1,y1},{x2,y2},...}]` задаются обе координаты точек.

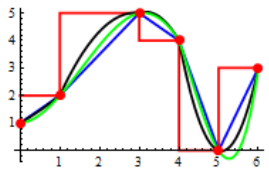
```
pnt = {{0, 1}, {1, 2}, {4, 4}, {3, 5}, {5, 0}, {6, 3}};
f = Interpolation[pnt]
f[2.5]
Plot[f[x], {x, 0, 6}, PlotStyle → {Black, Thickness[0.015]},
                Epilog–> {Red, PointSize[0.04], Point/@pnt}]
InterpolatingFunction[{{0,6}}," <> "]
4.59375
```



Обратите внимание, что абсциссы точек не обязаны следовать в возрастающем порядке.

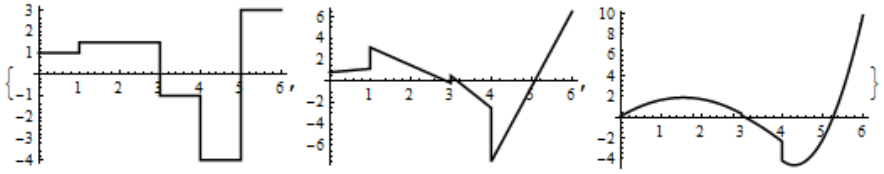
На участках между исходными точками интерполяционная функция представляется полиномами, степень которых можно задавать с помощью опции `InterpolationOrder`. В следующем коде на одном наборе точек мы создаем четыре функции с порядком интерполяции 0, 1, 2, 3 и строим их графики.

```
pnt = {{0, 1}, {1, 2}, {4, 4}, {3, 5}, {5, 0}, {6, 3}};
Table[ToExpression["f" <> ToString[i]][x_] =
     Interpolation[pnt, InterpolationOrder → i][x], {i, 0, 3}];
Plot[{f0[x], f1[x], f2[x], f3[x]}, {x, 0, 6},
    PlotStyle → {{Red, Thickness[0.01]}, {Blue, Thickness[0.01]},
                 {Black, Thickness[0.01]}, {Green, Thickness[0.01]}},
    Epilog–> {Red, PointSize[0.04], Point/@pnt}]
```



Нулевой порядок создает кусочно – постоянную интерполяционную функцию, первый – непрерывную кусочно – линейную и т.д. Однако это не сплайн – интерполяция. Это видно из графиков производных.

```
{Plot[Evaluate[D[f1[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}],
 Plot[Evaluate[D[f2[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}],
 Plot[Evaluate[D[f3[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}]}
```



Обычно при сплайн – интерполяции `n` – го порядка в точках стыка сегментов кусочного полинома выполняется непрерывность производных до порядка `n-1` включительно. Из графиков видно, что уже первая производная интерполяционных функций имеет разрывы в точках стыка. Ниже мы узнаем, что по умолчанию выполняется интерполяция Эрмита.

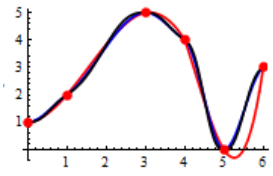
В связи со сказанным, имеется формат вызова функции `Interpolation`, в котором можно задавать значения производных в узловых точках. Этот формат имеет вид

`Interpolation[{{{x₁}, y₁, y₁′, ...}, {{x₂}, y₂, y₂′, ...}, ...}]`,

где x_i – абсцисса `i`-ой точки, y_i – ордината этой точки, y_i' – значение первой производной и т.д.

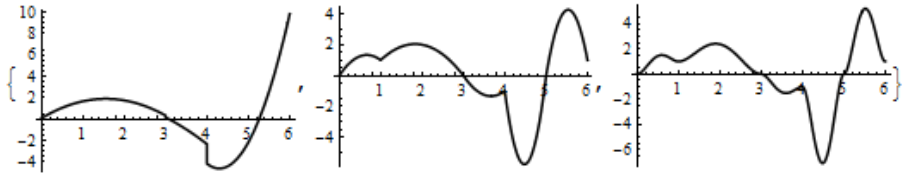
В следующем примере мы строим три интерполяционные функции 3 – го порядка на одном и том же наборе точек $\{x_i, y_i\}$. Для функции `f3` мы не задаем значение производных, для функции `g3` – задаем значения первых производных, для функции `h3` – задаем значения первых и вторых производных в узловых точках.

```
pnt0 = {{0, 1}, {1, 2}, {4, 4}, {3, 5}, {5, 0}, {6, 3}};
pnt1 = {{{0}, 1, 0}, {{1}, 2, 1}, {{4}, 4, −1}, {{3}, 5, 0}, {{5}, 0, 0}, {{6}, 3, 1}};
pnt2 = {{{0}, 1, 0, 0}, {{1}, 2, 1, 0}, {{4}, 4, −1, 0}, {{3}, 5, 0, 0},
                                                {{5}, 0, 0, 0}, {{6}, 3, 1, 0}};
f3 = Interpolation[pnt0, InterpolationOrder → 3]
g3 = Interpolation[pnt1, InterpolationOrder → 3]
h3 = Interpolation[pnt2, InterpolationOrder → 3]
Plot[{f3[x], g3[x], h3[x]}, {x, 0, 6}, PlotStyle → {{Red, Thickness[0.01]},
                    {Blue, Thickness[0.01]}, {Black, Thickness[0.01]}},
      Epilog−> {Red, PointSize[0.04], Point/@pnt}]
```



Как видим, кривые не очень отличаются. Однако, если построить графики первых производных этих функций, то различие существенное.

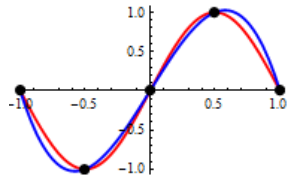
```
{Plot[Evaluate[D[f3[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}],
 Plot[Evaluate[D[g3[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}],
 Plot[Evaluate[D[h3[x], x]], {x, 0, 6}, PlotStyle → {Black, Thickness[0.01]}]}
```



Как видно, производная функции `f3` имеет разрыв, функция `g3'` непрерывна, но имеет резкие изломы, функция `h3'` уже гладкая.

Проверим качество интерполяции следующим примером, задав узлы на синусоиде.

```
ts = Chop[Table[{x, Sin[πx]}, {x, -1, 1, 0.5}]]
f = Interpolation[ts];
Plot[{Sin[πx], f[x]}, {x, -1, 1},
        PlotStyle → {{Red, Thickness[0.01]}, {Blue, Thickness[0.01]}},
        Epilog-> {Black, PointSize[0.04], Point/@ts}]
{{-1.,0},{-0.5,-1.},{0,0},{0.5,1.},{1.,0}}
```



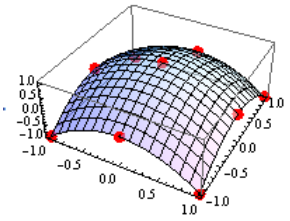
Если в этом примере шаг точек уменьшить, то графики будут сливаться.

Можно приближать многомерные данные. Например в двумерном случае формат вызова может быть следующим

`Interpolation[{{{x1, y1}, z1}, {{x2, y2}, z2}}, ...}],`

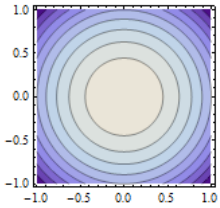
где $\{x_i, y_i\}$ – координаты точки на плоскости, а z_i – значение функции в этой точке.

```
data = Flatten[Table[{{x, y}, 1 - x^2 - y^2}, {x, -1, 1, 1}, {y, -1, 1, 1}], 1];
surf = Interpolation[data, InterpolationOrder → 2]
Show[Plot3D[surf[x, y], {x, -1, 1}, {y, -1, 1}, PlotStyle → Opacity[.5]],
     Graphics3D[{Red, PointSize[0.05],
                 Map[Point, Partition[Flatten[data], 3]]}]]
```



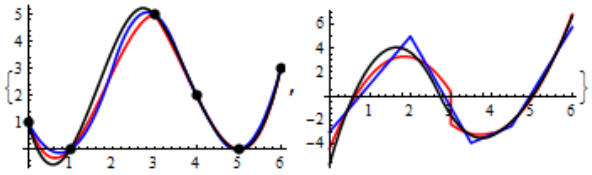
В многомерном случае, кроме значения функции в узлах можно задавать обе частные производные (т.е. в узле задавать градиент).

```
z[x_, y_] = 1 - x^2 - y^2;
data = Flatten[Table[Evaluate[{{x, y}, z[x, y], D[z[x, y], {{x, y}}]}],
{x, -1, 1, 0.5}, {y, -1, 1, 0.5}], 1]
 {{{-1.,-1.},-1.,{2.,2.}},{{-1.,-0.5},-0.25,{2.,1.}},...
f = Interpolation[data]
ContourPlot[f[x, y], {x, -1, 1}, {y, -1, 1}]
```



У функции `Interpolation` есть опция `Method`, которая задает метод полиномиальной интерполяции. Это может быть сплайн – интерполяция или интерполяция Эрмита. В следующем примере мы строим интерполяционные функции, используя оба метода, причем строим сплайны 2 – го и 3 – го порядков.

```
pnt0 = {{0, 1}, {1, 0}, {4, 2}, {3, 5}, {5, 0}, {6, 3}};
FH = Interpolation[pnt0, Method → "Hermite"];
FS2 = Interpolation[pnt0, Method → "Spline", InterpolationOrder → 2];
FS3 = Interpolation[pnt0, Method → "Spline", InterpolationOrder → 3];
{Plot[{FH[x], FS2[x], FS3[x]}, {x, 0, 6}, PlotStyle → {Red, Blue, Black},
      Epilog → {Black, PointSize[0.04], Point/@pnt0}],
Plot[Evaluate[{D[FH[x], x], D[FS2[x], x], D[FS3[x], x]}], {x, 0, 6},
      PlotStyle → {Red, Blue, Black}]}
```



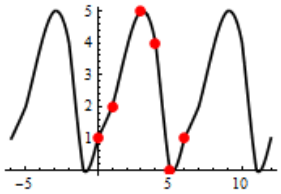
Здесь на графике слева показаны интерполяционные кривые, а справа – их производные. Из правого графика видно, что интерполяционная функция Эрмита **FH** имеет разрыв производной. Производная сплайна 2 – го порядка **FS2** непрерывна, но имеет изломы. Гладкость производной сплайна 3 – го порядка **FS3** говорит о том, что у него непрерывна 2 – я производная.

Использование сплайн интерполяции невозможно, когда в узлах задаются значения производных. Например следующий код выводит сообщение о том, что сплайн – интерполяция невозможна. После этого автоматически выполняется интерполяция Эрмита.

```
pnt1 = {{{0}, 1, 0}, {{1}, 0, 1}, {{4}, 2, −1}, {{3}, 5, 0}, {{5}, 0, 0}, {{6}, 3, 1}};
FS = Interpolation[pnt1, Method → Spline];
FS[2.5]
Interpolation::mspl: The Spline method could not be used because
the data could not be coerced to machine real numbers
4.3125
```

Имеется еще опция `PeriodicInterpolation`. Если она принимает значение `True`, то строится периодическая интерполирующая функция.

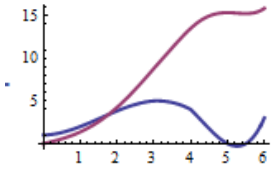
```
pnt0 = {{0, 1}, {1, 2}, {4, 4}, {3, 5}, {5, 0}, {6, 1}};
f = Interpolation[pnt0, PeriodicInterpolation → True];
Plot[f[x], {x, −6, 12}, PlotStyle → {Black, Thickness[0.01]},
                        Epilog−> {Red, PointSize[0.04], Point/@pnt0}]
```



При использовании этой опции следует помнить, что значения в первом и последнем узле должны задаваться одинаковыми.

Можно интегрировать интерполяционную функцию. В следующем коде мы строим графики интерполяционной кривой и ее первообразной.

$$\mathbf{pnt0} = \{\{0,1\},\{1,2\},\{4,4\},\{3,5\},\{5,0\},\{6,3\}\};$$
$$f = \mathbf{Interpolation}[\mathbf{pnt0}];$$
$$F[x\_] = \mathbf{Integrate}[f[x], x]$$
$$\mathbf{Plot}[\{f[x], F[x]\}, \{x, 0, 6\}, \mathbf{PlotStyle} \to \mathbf{Thickness}[0.01]]$$



Сделаем еще несколько замечаний о функции `Interpolation`.

Когда порядок интерполяции задан по умолчанию, нужно передавать не менее 4 – х узлов.

$$\mathbf{Interpolation}[\{\{0,1\},\{1,2\},\{4,4\}\}];$$

```
Interpolation::inhr: Requested order is too high; order has been
reduced to {2}.
```

Интерполяционная функция всегда создается непрерывной, но необязательно дифференцируемой.

Значения функции можно задавать комплексными, но координаты точек должны быть вещественными.

Элементы структуры объекта `InterpolatingFunction[{{1,6}},"<>"]` можно проанализировать с помощью индексации. Эти элементы обычно не отображаются в выводе, но они всегда присутствуют.

$$\mathbf{pnt0} = \{\{0,1\},\{1,2\},\{3,5\},\{4,4\},\{5,0\},\{6,1\}\};$$
$$f = \mathbf{Interpolation}[\mathbf{pnt0}, \mathbf{InterpolationOrder} \to 2];$$
$$\{f[[0]], f[[1]], f[[2]], f[[3]], f[[4]], f[[5]]\}$$

```
InterpolatingFunction[{{0,6}},<>]
{InterpolatingFunction,{{0,6}},
 {4,3,0,{6},{3},0,0,0,0,Automatic},
 {{0,1,3,4,5,6}},{{1},{2},{5},{4},{0},{1}},{Automatic}}
```

Элемент `f[[1]]` содержит диапазон `{{0,6}}`, на котором построена интерполяционная функция. Элемент `f[[3]]` содержит абсциссы узлов, а элемент `f[[4]]` содержит их ординаты (при интерполяции по Эрмиту). Меняя данные и значения опций, можно сделать вывод, что элемент `f[[2,4]]` хранит количество узлов, элемент `f[[2,5]]` на единицу больше порядка интерполяции `InterpolationOrder`.

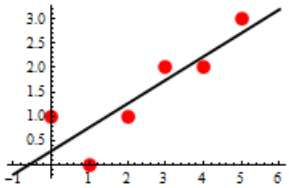
3.7.3 Аппроксимация

В *Mathematica* имеются функции, которые решают близкие к интерполяции по смыслу задачи.

Функция `Fit` выполняет приближение методом наименьших квадратов набора данных с помощью линейной комбинации заданного набора функций.

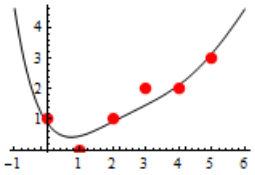
Например, пусть заданы точки и требуется приблизить их линейной функцией. Это можно сделать следующим образом.

```
data = {{0, 1}, {1, 0}, {2, 1}, {3, 2}, {4, 2}, {5, 3}};
line = line = Fit[data, {1, x}, x]
0.28571429 + 0.48571429 x
Show[Plot[line, {x, -1, 6}, PlotStyle → {Black, Thickness[0.01]}],
     Graphics[{Red, PointSize[0.05], Point[data]}]]
```



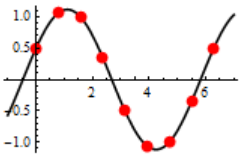
Здесь для приближения мы использовали две функции: 1 (единица) и x. Для приближения тех же данных можно использовать другие функции или больший набор «пробных» функций. В следующем примере мы используем четыре функции: 1 (единица), x, $\sin x$ и e^{-x}.

```
line = Fit[data, {1, x, Sin[x], Exp[-x]}, x]
Show[Plot[line, {x, -1, 6}, PlotStyle → {Black, Thickness[0.01]}],
     Graphics[{Red, PointSize[0.05], Point[data]}],
     AxesOrigin → {0, 0}, PlotRange → All]
-2.2151282+3.0802025 e^-x+1.1566815 x+0.45308887 Sin[x]
```



В следующем примере мы выбираем дискретный набор точек на кривой $\sin x + \frac{1}{2}\cos x$ и аппроксимируем его функциями $\sin x$ и $\cos x$.

```
data = Table[{x, Sin[x] + 1/2 Cos[x]}, {x, 0, 2π, π/4}]
line = Chop[Fit[data, {1, x, Sin[x], Cos[x]}, x]]
Show[Plot[line, {x, -1, 7}, PlotStyle → {Black, Thickness[0.01]}],
     Graphics[{Red, PointSize[0.05], Point[data]}],
     AxesOrigin → {0, 0}, PlotRange → All]
0.5 Cos[x]+1.0 Sin[x]
```



После отбрасывания погрешностей вычислений функцией `Chop` мы получили исходное выражение. Как видим, приближение очень хорошее.

Можно приближать множества точек в пространстве с помощью функций 2 – х переменных.

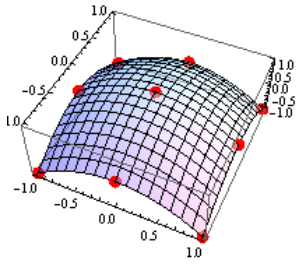
$\mathbf{data = Flatten[Table[\{x, y, 1 - x^2 - y^2\}, \{x, -1, 1, 1\}, \{y, -1, 1, 1\}], 1]}$

$\mathbf{surf = Chop[Fit[data, \{1, x, y, x^2, y^2, xy\}, \{x, y\}]]}$

$\{\{-1,-1,-1\},\{-1,0,0\},\{-1,1,-1\},\{0,-1,0\},$
$\{0,0,1\},\{0,1,0\},\{1,-1,-1\},\{1,0,0\},\{1,1,-1\}\}$

`1.-1.0 x^2-1.0 y^2`

$\mathbf{Show[Plot3D[surf, \{x, -1, 1\}, \{y, -1, 1\}, PlotStyle \to Opacity[.5]],}$
$\mathbf{Graphics3D[\{Red, PointSize[0.05], Map[Point, data]\}]]}$



С другими функциями системы близкими к рассмотренным по смыслу, вы можете познакомиться по справочной системе.

3.8 Решение прикладных задач.

Приложения математического анализа неисчислимы. Здесь мы выбрали несколько тем, которые демонстрируют возможности описанных в данной главе функций системы.

3.8.1 Физические приложения кратных интегралов.

Центр тяжести тела. Пусть $\rho = \rho(x, y)$ представляет плотность материала двумерного тела (масса единицы площади в точке *x*,*y*). Тогда масса тела, занимающего область *D* на плоскости, вычисляется по формуле

$$M = \iint_D \rho(x, y)\, d\,x\, d\,y \qquad (1)$$

Положение (x_c, y_c) центра тяжести тела определяется по формулам

$$x_c = \frac{1}{M}\iint_D x\,\rho(x, y)\, d\,x\, d\,y,\ \ y_c = \frac{1}{M}\iint_D y\,\rho(x, y)\, d\,x\, d\,y \qquad (2)$$

Когда говорят о центре тяжести фигуры, то подразумевают, что плотность $\rho(x, y) = 1$.

Пример. Найти центр тяжести первого квадранта эллипса. Имеем

$\mathbf{Clear[a, b]};$

$\mathbf{opt = Sequence[\{x, 0, a\}, \{y, 0, b\}, Assumptions -> a > 0\ \&\&\ b > 0]};$

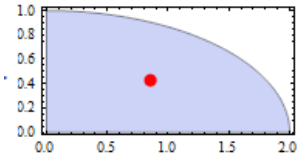
$\mathbf{rg = \frac{x^2}{a^2} + \frac{y^2}{b^2} \le 1\ \&\&\ x \ge 0\ \&\&\ y \ge 0};$

$\mathbf{M = Integrate[Boole[rg], opt]};$

```
Mx = Integrate[x Boole[rg], opt];
My = Integrate[y Boole[rg], opt];
{xc, yc} = {Mx, My}/M
```

$$\left\{\frac{4a}{3\pi}, \frac{4b}{3\pi}\right\}$$

```
a = 2; b = 1;
rg = RegionPlot[rg, {x, 0, a}, {y, 0, b}, AspectRatio → Automatic];
pt = Graphics[{Red, PointSize[0.05], Point[{xc, yc}]}];
Show[rg, pt]
```



□

Пример. Найти центр тяжести фигуры, ограниченной ветвью циклоиды $x = a(t - \sin t)$, $y = a(1 - \cos t)$ и осью x.

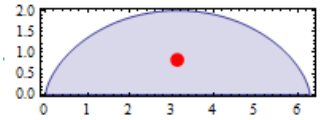
В формуле Грина $\iint\limits_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) d\,x\,d\,y = \oint\limits_L P\,d\,x + Q\,d\,y$ положим $Q = x^2/2, P = 0$. Тогда она принимает вид $\iint\limits_D x\,d\,x\,d\,y = \oint\limits_L \frac{x^2}{2} d\,y$, где L – контур, составленный из циклоиды и отрезка $[0, 2\pi]$ оси Ox, на котором $d\,y = 0$. Поэтому под контуром L можно понимать только дугу циклоиды. Аналогично, полагая $Q = 0, P = -y^2/2$, получаем $\iint\limits_D y\,d\,x\,d\,y = -\oint\limits_L \frac{y^2}{2} d\,x$, где L – контур, составленный из циклоиды и отрезка $[0, 2\pi]$ оси Ox, на котором $y = 0$ и, поэтому под контуром L можно понимать только одну дугу циклоиды. В результате формулы (2) принимают вид $x_c = \frac{1}{2M}\oint\limits_L x^2\,d\,y$ и $y_c = -\frac{1}{2M}\oint\limits_L y^2\,d\,x$, где M определяется как и раньше по (1). Заменяя x и y их параметрическим представлением, приходим к необходимым формулам вычисления центра тяжести нашей фигуры.

```
a =.; x = a(t - Sin[t]); y = a(1 - Cos[t]);
M = -Integrate[xD[y, t], {t, 0, 2π}];
Mx = -Integrate[x^2 D[y, t], {t, 0, 2π}]/2;
My = -Integrate[-y^2 D[x, t], {t, 0, 2π}]/2;
{xc, yc} = {Mx, My}/M
```

$$\left\{a\,\pi, \frac{5a}{6}\right\}$$

Обратите внимание на минусы, которые необходимо поставить перед выражениями для *M*, *Mx* и *My*, поскольку циклоида обходит область в направлении по часовой стрелки, т.е. в отрицательном направлении.

```
a = 1;
rg = ParametricPlot[{vx, vy}, {t, 0, 2π}, {v, 0, 1}, Mesh → None];
pt = Graphics[{Red, PointSize[0.05], Point[{xc, yc}]}];
Show[rg, pt]
```



□

Положение центра тяжести трехмерного тела определяется по формулам

$$x_c = \frac{1}{M}\iiint_V \rho\, x\, dx\, dy\, dz,\ y_c = \frac{1}{M}\iiint_V \rho\, y\, dx\, dy\, dz,\ z_c = \frac{1}{M}\iiint_V \rho\, z\, dx\, dy\, dz,$$

где $M = \iiint_V \rho\, dx\, dy\, dz$ представляет массу тела.

Пример. Найдем центр массы полушара $x^2 + y^2 + z^2 \le 1\ \ (z \ge 0)$ плотности $\rho = 1$. В силу симметрии $\iiint_V x\, dx\, dy\, dz = 0$ и $\iiint_V y\, dx\, dy\, dz = 0$. Далее имеем

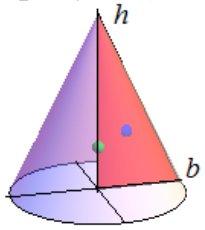
```
Mz = Integrate[z Boole[x^2 + y^2 + z^2 ≤ 1 && z ≥ 0],
                {x, -1, 1}, {y, -1, 1}, {z, 0, 1}]
M = Integrate[Boole[x^2 + y^2 + z^2 ≤ 1 && z ≥ 0],
                {x, -1, 1}, {y, -1, 1}, {z, 0, 1}]
zc = Mz/M
```

$$\frac{3}{8}$$

Т.о. центр массы единичного полушара находится в точке $\left(0, 0, \frac{3}{8}\right)$.

□

Пример. Определить положение центра тяжести плоского треугольника, и конуса, образованного вращением этого треугольника вокруг одного из катетов на оси *Oz*. Вершина треугольника находится в точке (0,0,*h*), а длина катета основания равна *b* (см. следующий рисунок)



Вычисляем положение центра тяжести треугольника

```
Clear[h, b, x, y, z];
brg = Boole[y ≤ h (1 - x/b) && x ≥ 0 && y ≥ 0];
opt = Sequence[{x, 0, b}, {y, 0, h}, Assumptions :→ b > 0 && h > 0];
M = Integrate[brg, opt];
Mx = Integrate[x * brg, opt];
My = Integrate[y * brg, opt];
```

$\{\mathbf{xc}, \mathbf{zc}\} = \{\mathbf{Mx}, \mathbf{My}\}/\boldsymbol{M}$

$\left\{\frac{b}{3}, \frac{h}{3}\right\}$

Вычисляем положение центра тяжести конуса. В силу симметрии его центр тяжести лежит на оси Oz.

$\mathbf{crg} = \mathbf{Boole}[\mathbf{0} <= \boldsymbol{z} \leq \boldsymbol{h}(\mathbf{1} - \frac{\sqrt{\boldsymbol{x^2} + \boldsymbol{y^2}}}{\boldsymbol{b}})];$

$\mathbf{opt} = \mathbf{Sequence}[\{\boldsymbol{x}, -\infty, \infty\}, \{\boldsymbol{y}, -\infty, \infty\}, \{\boldsymbol{z}, \mathbf{0}, \boldsymbol{h}\},$
$\mathbf{Assumptions} :\rightarrow \boldsymbol{b} > 0 \;\&\&\; \boldsymbol{h} > 0];$

$\mathbf{MC} = \mathbf{Integrate}[\mathbf{crg}, \mathbf{opt}];$

$\mathbf{MCz} = \mathbf{Integrate}[\boldsymbol{z} * \mathbf{crg}, \mathbf{opt}];$

$\mathbf{Zc} = \mathbf{MCz}/\mathbf{MC}$

$\frac{h}{4}$

Как видим вертикальная координата центра тяжести конуса отличается от вертикальной координаты центра тяжести треугольника.

Нарисуем наши фигуры и положения их центров тяжести (см. предыдущий рисунок)

$\boldsymbol{h} = \mathbf{2}; \boldsymbol{b} = \mathbf{1};$

$\mathbf{ps} = \mathbf{Plot3D}[\boldsymbol{h}(\mathbf{1} - \frac{\sqrt{\boldsymbol{x^2} + \boldsymbol{y^2}}}{\boldsymbol{b}}), \{\boldsymbol{x}, -\boldsymbol{b}, \boldsymbol{b}\}, \{\boldsymbol{y}, -\boldsymbol{b}, \boldsymbol{b}\},$
$\mathbf{RegionFunction} \rightarrow \mathbf{Function}[\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}, \boldsymbol{x^2} + \boldsymbol{y^2} \leq \boldsymbol{b^2}],$
$\mathbf{BoxRatios} \rightarrow \{\mathbf{2}\boldsymbol{b}, \mathbf{2}\boldsymbol{b}, \boldsymbol{h}\}, \mathbf{PlotStyle} \rightarrow \mathbf{Opacity}[\mathbf{0.5}], \mathbf{Mesh} \rightarrow \mathbf{None}];$

$\mathbf{pt} = \mathbf{Graphics3D}[\{\mathbf{Red}, \mathbf{Polygon}[\{\{\mathbf{0}, \mathbf{0}, \mathbf{0}\}, \{\boldsymbol{b}, \mathbf{0}, \mathbf{0}\}, \{\mathbf{0}, \mathbf{0}, \boldsymbol{h}\}\}],$
$\mathbf{Blue}, \mathbf{Sphere}[\{\mathbf{xc}, \mathbf{0}, \mathbf{zc}\}, \mathbf{0.07}],$
$\mathbf{Green}, \mathbf{Sphere}[\{\mathbf{0}, \mathbf{0}, \mathbf{Zc}\}, \mathbf{0.07}]\}];$

$\mathbf{Show}[\mathbf{ps}, \mathbf{pt}, \mathbf{Boxed} \rightarrow \mathbf{False}, \mathbf{AxesOrigin} \rightarrow \{\mathbf{0}, \mathbf{0}, \mathbf{0}\}, \mathbf{Ticks} \rightarrow \mathbf{None}]$

□

Механическая работа.

Работа при перемещении тела в силовом поле $\mathbf{F} = \big(P(x, y, z), Q(x, y, z), R(x, y, z)\big)$ вдоль кривой L выражается криволинейным интегралом

$$A = \oint_L \mathbf{F} \cdot d\,\mathbf{r} = \oint_L P\,d\,x + Q\,d\,y + R\,d\,z$$

При движении в плоскости z=0 третье слагаемое в криволинейном интеграле отсутствует. Если траектория движения L задана параметрически $x = x(t),\ y = y(t)$, то формула принимает вид

$$A = \int_{t_0}^{t_1} \big(P(x(t), y(t))x'(t) + Q(x(t), y(t))y'(t)\big)d\,t\,,$$

где t_0 и t_1 определяют положение начальной и конечной точек кривой.

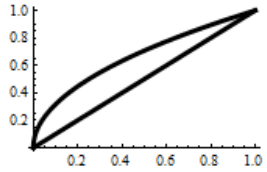
Пример. Найти работу поля $\mathbf{F} = \big(x\,y, x + y\big)$ при перемещении тела из начала координат $O(0,0)$ в точку $A(1,1)$ по кривой L_1 (отрезок прямой $y = x$) и по кривой L_2 (отрезок параболы $y^2 = x$).

```
y1[x_] = x;
y2[x_] = √x;
Work = Integrate[x * #, {x, 0, 1}] + Integrate[(x + #)D[#, x], {x, 0, 1}]&;
Work[y1[x]]
Work[y2[x]]
```

$$\frac{4}{3}$$

$$\frac{37}{30}$$

```
Plot[{y1[x], y2[x]}, {x, 0, 1}, PlotStyle → {{Black, Thickness[0.02]}}]
```



□

Момент инерции.

Моментом инерции материальной точки относительно оси называется произведение ее массы на квадрат расстояния точки до этой оси.

Для двумерных тел плотностью $\rho(x, y)$ момент инерции относительно осей O*x* и O*y* вычисляется по формулам

$$Ix = \iint\limits_D \rho(x, y) y^2 \, dx \, dy, \quad Iy = \iint\limits_D \rho(x, y) x^2 \, dx \, dy$$

При вычислении моментов инерции геометрических фигур полагаем $\rho = 1$.

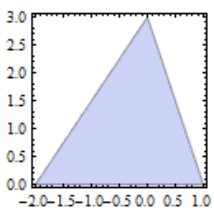
Пример. Найти момент инерции треугольника с основанием b и высотой h относительно его основания.

Основание треугольника примем за ось O*x*, а его высоту – за ось O*y*. Абсциссы точек пересечения боковых сторон с осью O*x* обозначим b_1 и b_2 $(b = b_2 - b_1)$. Имеем

```
Clear[b1, b2, b, h];
Ix0 = Integrate[y^2 Boole[x/b1 + y/h ≤ 1 && x/b2 + y/h ≤ 1 && y ≥ 0],
    {x, -∞, ∞}, {y, -∞, ∞}, Assumptions :→ h > 0 && b1 < 0 && b2 > 0];
Ix = Simplify[Ix0]/.(b2 - b1) → b
```

$$\frac{bh^3}{12}$$

```
b1 = -2; b2 = 1; h = 3;
RegionPlot[x/b1 + y/h ≤ 1 && x/b2 + y/h <= 1 && y ≥ 0, {x, b1, b2}, {y, 0, h}]
```



Пример. Найти момент инерции эллипса $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$ относительно его главных осей.

```
Clear[x, y, a, b];
Ix = Integrate[y^2 Boole[x^2/a^2 + y^2/b^2 ≤ 1], {x, -∞, ∞}, {y, -∞, ∞},
                                    Assumptions :→ a > 0 && b > 0]
Iy = Integrate[x^2 Boole[x^2/a^2 + y^2/b^2 ≤ 1], {x, -∞, ∞}, {y, -∞, ∞},
                                    Assumptions :→ a > 0 && b > 0]
```

$\frac{1}{4} a\, b^3\, \pi$

$\frac{1}{4} a^3\, b\, \pi$

□

3.8.2 Примеры решения задач распространения тепла

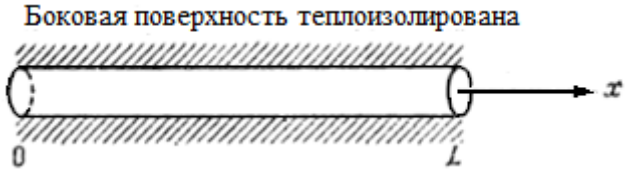
Задача о распространении тепла в тонком однородном неограниченном стержне, боковая поверхность которого теплоизолирована (тепло может распространяться только вдоль оси x), математически формулируется следующим образом [3, 7]: найти ограниченную функцию $u(x,t)$ $(t>0, -\infty<x<\infty)$, удовлетворяющую уравнению теплопроводности

$$\frac{\partial u}{\partial t}=a^2\frac{\partial^2 u}{\partial x^2} \qquad (t>0, -\infty<x<\infty) \tag{1}$$

и начальному условию

$$u(x,0)=\varphi(x) \qquad (-\infty<x<\infty). \tag{2}$$

где $u(x,t)$ температура сечения x стержня в момент времени t (стержень тонкий, это значит, что температура точек любого поперечного сечения x одинакова).



Известно, что общее решение этой задачи представляется интегралом Пуассона следующего вида

$$u(x,t)=\frac{1}{2a\sqrt{\pi t}}\int_{-\infty}^{\infty}\varphi(\xi)e^{-\frac{(\xi-x)^2}{4a^2t}}d\xi \tag{3}$$

Если в этом интеграле сделать замену $\tau=\frac{\xi-x}{2a\sqrt{t}}$, то мы получим

$$u(x,t)=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}\varphi\left(x+2a\sqrt{t}\,\tau\right)e^{-\tau^2}d\tau \tag{4}$$

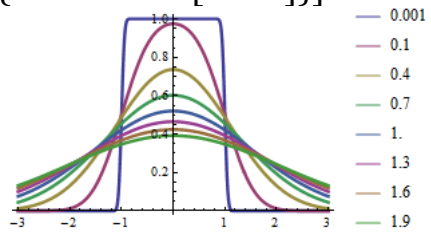
Пример. *Бесконечный стержень. Начальная температура всюду ноль, кроме отрезка* [-1,1], *где она равна единице*, т.е. $u(x,0)=\varphi(x)=\begin{cases}1, -1\le x\le 1\\ 0, x<-1\vee x>1\end{cases}$. В этом примере интеграл (4) вычисляется символьно и выражается через интеграл ошибок.

```
a =.;
φ[x_] = Piecewise[{{0, x ≤ -1}, {1, x < 1}}, 0];
u[x_, t_] = FullSimplify[1/Sqrt[π] Integrate[φ[x + 2a Sqrt[t] τ] Exp[-τ^2],
                {τ, -∞, ∞}, Assumptions :→ t > 0 && a > 0]]
```

$$\frac{1}{2}\left(\text{Erf}\left[\frac{1-x}{2a\sqrt{t}}\right]+\text{Erf}\left[\frac{1+x}{2a\sqrt{t}}\right]\right)$$

На следующем рисунке представлены графики температуры $u(x,t)$ в различные моменты времени (коэффициент теплопроводности $a^2=1$).

```
T = Join[{0.001}, Table[t, {t, 0.1, 2, 0.3}]]; a = 1;
Plot[Evaluate[Table[u[x, t], {t, T}]], {x, -3, 3}, PlotLegends → {T},
     PlotStyle → {Thickness[0.01]}]
```



□

Рассмотрим задачу о распространении тепла в полуограниченном стержне, боковая поверхность которого теплоизолирована, а конец x=0 поддерживается при нулевой температуре, т.е. $u(0,t)=0\ \ (t\ge 0)$. Решение этой задачи может быть получено методом продолжения, когда начальная температура $\varphi(x)$, заданная на положительной полуоси Ox, продолжается на отрицательную полуось нечетным образом. Можно показать [3], что решение в форме интеграла Пуассона (3) с любой нечетной функцией $\varphi(\xi)$ будет представлять решение задачи (1), (2), которое обращается в ноль при x=0. Оно, тем самым, при положительных x будет давать решение поставленной задачи для полубесконечного стержня.

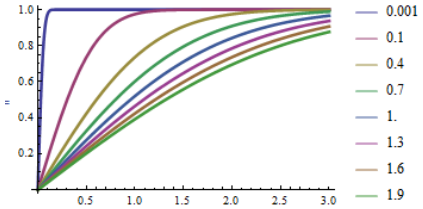
Пример. *Полубесконечный стержень. Начальная температура всюду равна 1, температура на левом конце равна нулю,* т.е. $u(x,0)=1\ \ (x\ge 0)$, $u(0,t)=0\ \ (t\ge 0)$.

Вначале строим нечетную функцию $\varphi(x)=\begin{cases}1, x\ge 0\\ -1, x<0\end{cases}$ и подставляем ее в интеграл (4), который выражается интегралом ошибок. Затем строим графики функции $u(x,t)\ \ (x\ge 0)$ в некоторые фиксированные моменты времени.

```
φ[x_] = Piecewise[{{-1, x ≤ 0}}, 1]; a =.;
```

$$u[x\_, t\_] = \text{FullSimplify}[\frac{1}{\sqrt{\pi}}\text{Integrate}[\varphi[x + 2a\sqrt{t}\tau]\text{Exp}[-\tau^2],$$
$$\{\tau, -\infty, \infty\}, \text{Assumptions} :\to t > 0 \&\& a > 0]]$$
$$T = \text{Join}[\{0.001\}, \text{Table}[t, \{t, 0.1, 2, 0.3\}]]; a = 1;$$
$$\text{Plot}[\text{Evaluate}[\text{Table}[u[x, t], \{t, T\}]], \{x, 0, 3\}, \text{PlotLegends} \to \{T\},$$
$$\text{PlotStyle} \to \{\text{Thickness}[0.01]\}]$$

$$\text{Erf}\left[\frac{x}{2a\sqrt{t}}\right]$$



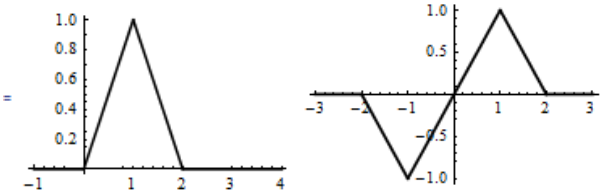
Результирующая последовательность графиков иллюстрирует процесс остывания полубесконечного стержня, левая граница которого поддерживается при нулевой температуре.

□

Пример. *Полубесконечный стержень. Начальная температура везде 0, кроме отрезка у границы, где она имеет треугольную форму. Температура левого конца равна нулю*, т.е. $u(0,t)=0 \ (t \ge 0)$.

В следующем коде мы строим график функции $\varphi(x)$, которая представляет начальную температуру (следующий рисунок слева) и график функции $\Phi(x)$, которая является ее нечетным продолжением (следующий рисунок справа).

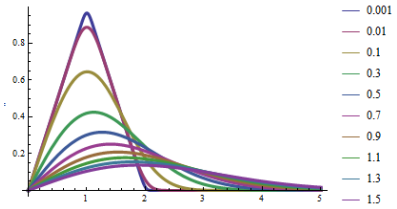
$$\varphi[x\_] = \text{Piecewise}[\{\{0, x \le 0\}, \{x, x < 1\}, \{2 - x, x < 2\}\}, 0];$$
$$\text{Plot}[\varphi[x], \{x, -1, 4\}] \qquad (* \text{ график слева } *)$$
$$\Phi[x\_] = \text{Piecewise}[\{\{0, x \le -2\}, \{-2 - x, x < -1\}, \{x, x < 1\},$$
$$\{2 - x, x < 2\}\}, 0];$$
$$\text{Plot}[\Phi[x], \{x, -3, 3\}] \qquad (* \text{ график справа } *)$$



Функцию $\Phi(x)$ используем в интеграле (4) для построения решения задачи. Интеграл вычисляется явно, но из – за длины мы не приводим полный вид его выражения.

$$a =.;$$
$$u[x\_, t\_] = \text{FullSimplify}[\frac{1}{\sqrt{\pi}}\text{Integrate}[\Phi[x + 2a\sqrt{t}\tau]\text{Exp}[-\tau^2],$$
$$\{\tau, -\infty, \infty\}, \text{Assumptions} :\to t > 0 \ \&\& \ a > 0]]$$
$$\frac{1}{2\sqrt{\pi}} e^{-\frac{(2+x)^2}{4a^2t}}(2a(-1 + e^{\frac{x}{a^2t}})(1 + e^{\frac{x}{a^2t}} - 2e^{\frac{3+2x}{4a^2t}})\sqrt{t} + \cdots)$$

```
T = Join[{0.001, 0.01}, Table[t, {t, 0.1, 1.5, 0.2}]];
a = 1;
Plot[Evaluate[Table[u[x, t], {t, T}]], {x, 0, 5},
                PlotStyle → {Thickness[0.01]}, PlotLegends → {T}]
```



□

Пусть начальная температура полубесконечного стержня равна 0 $(u(x,0)=0)$, а на его левом конце задана температура $u(0,t)=\psi(t)$. Решение этой задачи имеет вид ([2], ф.28 стр. 541)

$$u(x,t)=\frac{x}{2a\sqrt{\pi}}\int_0^t \frac{\psi(\tau)}{(t-\tau)^{\frac{3}{2}}} e^{-\frac{x^2}{4a^2(t-\tau)}} d\tau$$

Сделав замену переменной интегрирования $\xi=\dfrac{x}{2a\sqrt{t-\tau}}$, выражение для функции u приведем к следующему удобному для вычислений виду

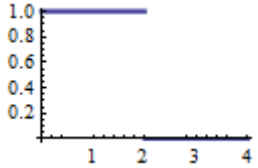
$$u(x,t)=\frac{1}{\sqrt{\pi}}\int_{\frac{x}{2a\sqrt{t}}}^{\infty} \psi(t-\frac{x^2}{4a^2\xi^2}) e^{-\xi^2} d\xi \qquad (5)$$

При вычислениях интегралов по этой формуле следует обращать внимание на особенность решения при t=0. Но формула «работоспособна» для любых $t>\tau>0$, где τ может быть сколь угодно малыми положительным числом.

Пример. *Полубесконечный стержень. Начальная температура всюду 0, температура на левом конце равна* $u(0,t)=\psi(t)=\begin{cases}1, 0\le t\le 2\\ 0, t>2\end{cases}$.

Вначале создаем функцию $\psi(t)$, представляющую граничный режим.

```
ψ[t_] = Piecewise[{{0, t < 0}, {1, t ≤ 2}}, 0];  a =.;
Plot[ψ[t], {t, 0, 4}]
```



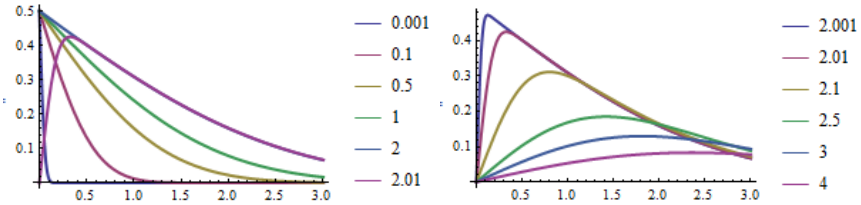
Теперь строим решение, используя интеграл (5).

```
u[x_, t_] = FullSimplify[1/Sqrt[π] Integrate[ψ[t - x^2/(4a^2 ξ^2)] Exp[-ξ^2], {ξ, x/(2a Sqrt[t]), ∞},
                Assumptions :→ t > 0 && a > 0 && x > 0]]
```

$$\begin{cases} \frac{1}{2}\left(\mathrm{Erf}\left[\frac{x}{2a\sqrt{-2+t}}\right]-\mathrm{Erf}\left[\frac{x}{2a\sqrt{t}}\right]\right) & a>0 \;\&\&\; t>2 \;\&\&\; x>0 \\ \frac{1}{2}\mathrm{Erfc}\left[\frac{x}{2a\sqrt{t}}\right] & \text{True} \end{cases}$$

Решение получено в форме кусочной функции. Теперь построим графики решения для моментов времени $t \le 2$ и для моментов $t > 2$.

```
T = {0.001, 0.1, 0.5, 1, 2, 2.01}; a = 1;
Plot[Evaluate[Table[u[x, t], {t, T}]], {x, 0, 3}, PlotLegends → {T},
     PlotStyle → {Thickness[0.01]}]   (* левый рисунок *)
T = {2.001, 2.01, 2.1, 2.5, 3, 4};
Plot[Evaluate[Table[u[x, t], {t, T}]], {x, 0, 3}, PlotLegends → {T},
     PlotStyle → {Thickness[0.01]}]   (* правый рисунок *)
```



Скачкообразное изменение граничной температуры в момент времени *t=2* меняет качественную картину распределения температуры у торца стержня после этого момента.

□

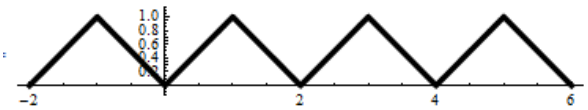
Пример. *Полубесконечный стержень. Начальная температура всюду 0, температура на левом конце периодически меняется по треугольному закону.*

Определим периодическую пилообразную непрерывную кусочно-линейную функцию

$$stx(x,w)=\left|x-w\left[\frac{x}{w}+\frac{1}{2}\right]\right| \quad \text{или} \quad stc(x,w)=\frac{w}{2\pi}\arccos\left(\cos\frac{2\pi x}{w}\right), \qquad (6)$$

где w является периодом этой функции, а квадратные скобки [x] обозначают функцию взятия наибольшего целого, не превосходящего *x*,

```
stc[x_, w_] = w/(2π) ArcCos[Cos[(2πx)/w]];
Plot[stc[x, 2], {x, -2, 6}, AspectRatio → Automatic]
```



Используем ее в качестве граничного значения $u(0,t)=\psi(t)=stc(t,2)$. Поскольку аналитически интеграл (5) не вычисляется, будем его считать численно.

```
a = 1; ψ[t_] := stc[t, 2];
u[x_, t_] := 2/√π NIntegrate[ψ[t - x^2/(4a^2 ξ^2)] Exp[-ξ^2], {ξ, x/(2a√t), ∞},
                AccuracyGoal → 4, PrecisionGoal → 4];
```

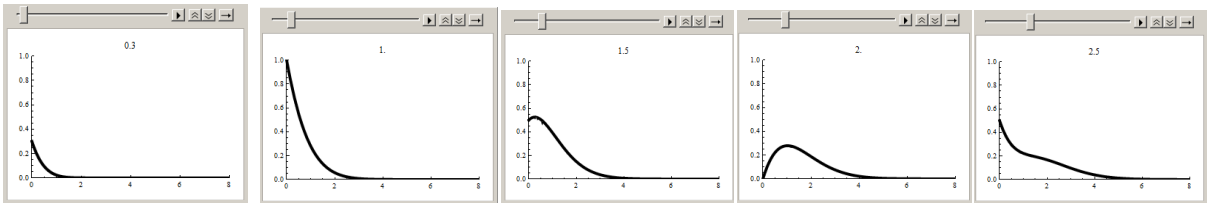
Для построения анимации используем функцию `ListAnimate`. Вначале создаем список графиков **lt**, которые будут представлять кадры анимации. Имейте ввиду, что создание списка **lt** займет некоторое время.

```
lt = Table[Plot[u[x, t], {x, 0, 8}, PlotStyle → {Black, Thickness[0.01]},
                PlotRange → {{0, 8}, {0, 1}}], {t, 0.1, 8, 0.1}];
```

Теперь строим анимацию.

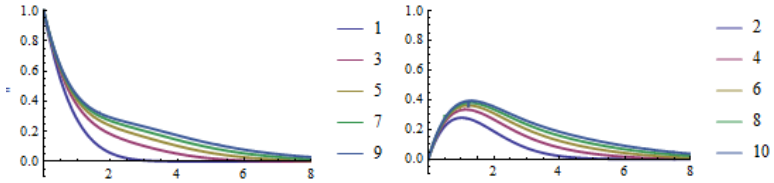
```
ListAnimate[lt, AnimationRunning → False]
```

На следующем рисунке представлено несколько кадров анимации – графиков распределения температуры в полубесконечном стержне в моменты времени $t = 0.3, 1.0, 1.5, 2.0, 2.5$



Графики распределения температуры внутри стержня через интервалы времени $\Delta t = 2$ имеют похожую форму. На следующем рисунке слева приведены графики температуры в моменты t = 1,3,5,7,9 и на рисунке справа – в моменты t = 2,4,6,8,10.

```
Plot[{u[x, 1], u[x, 3], u[x, 5], u[x, 7], u[x, 9]}, {x, 0, 8},
    PlotRange → {{0, 8}, {-0.1, 1}}, PlotLegends → {1, 3, 5, 7, 9}]
Plot[{u[x, 2], u[x, 4], u[x, 6], u[x, 8], u[x, 10]}, {x, 0, 8},
     PlotRange → {{0, 8}, {-0.1, 1}}, PlotLegends → {2, 4, 6, 8, 10}]
```



Графики в более поздние моменты времени имеют более высокий профиль.

□

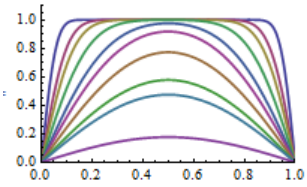
Решение уравнения теплопроводности для стержня конечной длины l, когда начальная температура задана функцией φ(ξ), а концы стержня поддерживаются при нулевой температуре, можно находить по формуле [5]

$$u(x,t) = \frac{1}{2l}\int_0^l \varphi(\xi)\left(\Theta_3\left(\pi\frac{\xi - x}{2l}, e^{-\frac{\pi^2 a^2 t}{l^2}}\right) - \Theta_3\left(\pi\frac{\xi + x}{2l}, e^{-\frac{\pi^2 a^2 t}{l^2}}\right)\right)d\xi, \qquad (7)$$

где $\Theta_3(z,q)$ является тета функция 3-го порядка ([4] стр.334). В пакете *Mathematica* она реализуется функцией `EllipticTheta[3,z,q]`.

Пример. *Конечный стержень длины l. Температура концов ноль. Начальная температура единица.* Вычисления интеграла по формуле (7) реализуем численно.

```
l = 1; a = 1; φ[x_] := 1;
θ3m[τ_, x_, t_] := EllipticTheta[3, π (τ - x)/(2 l), e^(-π^2 a^2 t/l^2)]
θ3p[τ_, x_, t_] := EllipticTheta[3, π (τ + x)/(2 l), e^(-π^2 a^2 t/l^2)]
u[x_, t_] := 1/(2 l) NIntegrate[φ[τ](θ3m[τ, x, t] - θ3p[τ, x, t]), {τ, 0, l},
                AccuracyGoal → 6, PrecisionGoal → 6]
Plot[{u[x, 0.001], u[x, 0.003], u[x, 0.005], u[x, 0.01], u[x, 0.02],
  u[x, 0.03], u[x, 0.05], u[x, 0.08], u[x, 0.1], u[x, 0.2]}, {x, 0, l},
  PlotRange → {{0, 1}, {0, 1.1}}, PlotStyle → {Thickness[0.01]}]
```

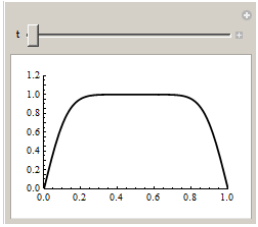


Для решения уравнения теплопроводности на конечном отрезке можно использовать классический метод Фурье [3]. Для рассмотренной нами задачи, когда начальная температура постоянна $\varphi(x)=u_0$, а на обеих границах поддерживается нулевая температура, решение Фурье имеет вид ([6], стр. 54)

$$u=\frac{4u_0}{\pi}\sum_{n=0}^{\infty}\frac{1}{(2n+1)}\sin\left(\frac{(2n+1)\pi x}{l}\right)\exp\left(-\frac{a(2n+1)^2\pi^2 t}{l^2}\right)$$

Построим манипулятор с графиками температуры в стержне в различные моменты времени.

```
u0 = 1; L = 1; a = 1;
u[t_, x_] := (4 u0)/π *
    Sum[1/(2n + 1) Sin[((2n + 1) π x)/L] Exp[-(a (2n + 1)^2 π^2 t)/L^2], {n, 0, 30}]
Manipulate[
  Plot[u[t, x], {x, 0, L}, PlotRange → {{0, L}, {0, 1.1}}],
  {t, 0.0001, 0.5}]
```



Обратите внимание, что для ускорения вычислений мы использовали частичную сумму ряда, представляющего решение (суммировали до *n*=30).

□

Пример. *Конечный стержень длины l. Начальная температура равна нулю. На границах поддерживаются постоянные температуры* u_1 *и* u_2.

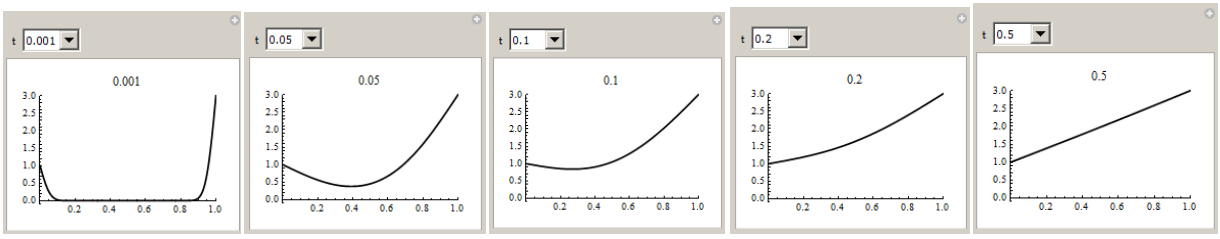
Для поставленной задачи решение Фурье имеет вид ([7], стр. 98, 101, 106)

$$u = u_1 + (u_2 - u_1)\frac{x}{l} + \frac{2}{\pi}\sum_{n=1}^{\infty}\frac{(-1)^n u_2 - u_1}{n}\sin\left(\frac{n\pi x}{l}\right)\exp\left(-\frac{a n^2 \pi^2 t}{l^2}\right)$$

Представим решение в различные моменты времени, используя манипулятор.

```
L = 1; a = 1; w1 = 1; w2 = 3;
T = Sequence[Join[{0.001}, Table[0.01 * i, {i, 50}]]];
DynamicModule[{u},
  u[t_, x_] := w1 + (w2 - w1) x/L +
    2/π Sum[((-1)^n w2 - w1)/n Sin[n π x/L] Exp[-a n^2 π^2 t/L^2], {n, 1, 50}];
  Manipulate[Plot[u[t, x], {x, 0, L},
    PlotRange -> {{0, L}, {-0.1, w2}}, PlotLabel → t], {t, T}]]
```

На следующем рисунке представлен манипулятор с графиками решения в моменты времени $t = 0.001, 0.05, 0.1, 0.2, 0.5$.



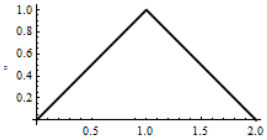
Для ускорения вычислений мы использовали частичную сумму ряда с максимальным значением n=50.

В моменты времени $t \geq 0.4$ графики распределения температуры практически не изменяются и совпадают с последним графиком. Это значит, что в стержне установилась температура $u(x,t) = u_1 + \frac{u_2 - u_1}{l}x$ и начиная с момента t=0.4 влияние начального условия пренебрежимо мало.

□

Пример. *Конечный стержень длины l. Температура концов ноль. Начальная температура имеет форму треугольника* $\varphi(x) = 1 - |x - 1|$.

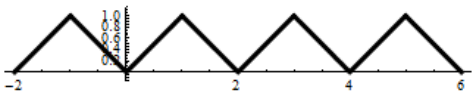
```
φ0[x_] = 1 - Abs[x - 1];
Plot[φ0[x], {x, 0, 2}, AspectRatio → Automatic]
```



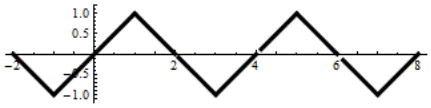
В этом примере мы используем решение для бесконечного стержня (4), используя метод продолжения. Нам надо построить нечетное периодическое продолжение начальной функции $\varphi(x)$ с отрезка [0,2] на всю вещественную ось. Для этого используем функцию $stc(x,w)$, определяемую формулой (6).

Если L длина отрезка, то соответствующее продолжение будет иметь вид $\varphi(x)=(-1)^{[x/L]}stc(x,L)$.

```
stc[x_, w_] = Abs[x - w Floor[x/w + 1/2]];
Plot[stc[x, 2], {x, -2, 6}, AspectRatio → Automatic]
```



```
φ[x_] = (-1)^Floor[x/2] stc[x, 2];
Plot[φ[x], {x, -2, 8}, AspectRatio → Automatic]
```

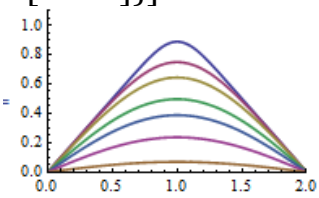


Подставив полученную функцию в (4), мы получим решение для конечного отрезка. Строим решение

```
a = 1;
u[x_, t_] := 1/Sqrt[π] NIntegrate[φ[x + 2 a Sqrt[t] τ] Exp[-τ^2], {τ, -∞, ∞},
                AccuracyGoal → 4, PrecisionGoal → 4,
                Method → "ClenshawCurtisRule"]
```

и его графики в разные моменты времени

```
Plot[{u[x, 0.01], u[x, 0.05], u[x, 0.1], u[x, 0.2], u[x, 0.3],
    u[x, 0.5], u[x, 1]}, {x, 0, 2}, PlotRange → {{0, 2}, {0, 1.1}},
    PlotStyle → {Thickness[0.01]}]
```

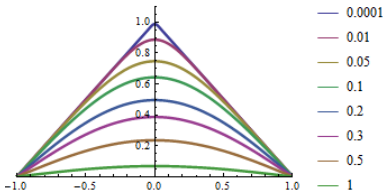


В работе [7] стр. 103 приводится решение в рядах для области $-l\le x\le l$ с начальной температурой $u_0(l-|x|/l)$ и нулевой температурой концов

$$u(x,t)=\frac{8u_0}{\pi^2}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^2}\cos\frac{(2n+1)\pi x}{2l}\exp\left(-\frac{a(2n+1)^2\pi^2 t}{4l^2}\right)$$

Используем его для сравнения с решением, полученным выше.

```
u0 = 1; L = 1; a = 1;
u[x_, t_] :=
  8 u0/π^2 Sum[1/(2n + 1)^2 Cos[(2n + 1) π x/(2L)] Exp[-a(2n + 1)^2 π^2 t/(4L^2)], {n, 0, 30}]
T = {0.0001, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 1};
Plot[Evaluate[Table[u[x, t], {t, T}]], {x, -1, 1},
    PlotRange → {{-1, 1}, {0, 1.1}}, PlotStyle → {Thickness[0.01]},
    PlotLegends → T]
```



Для ускорения вычислений мы использовали частичную сумму ряда, представляющего решение (суммировали до n=30).

Когда решение в виде ряда известно, то вычисления происходят значительно быстрее.

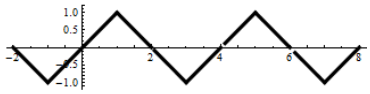
□

Пример. *Конечный стержень длины L. Температура левого конца ноль. Правый конец теплоизолирован. Начальная температура линейная функция.* Условие теплоизолированности правого конца математически формулируются как $\dfrac{\partial u(L,t)}{\partial x}=0$. Линейность начальной функции означает, что $u(x,0)=k\,x$.

Продолжим начальную функцию четно относительно точки $x=L$ на отрезок $[L,2L]$, затем полученную функцию с отрезка $[0,2L]$ продолжим нечетно на отрезок $[-2L,0]$, и затем с отрезка $[-2L,2L]$ продолжим периодически на всю ось. Описанное построение в п. 2.4.5 (пример 5.5) мы назвали четно – нечетным периодическим продолжением функции. Если полученную функцию $\Phi(x)$ подставить в (4), то получим решение нашей краевой задачи.

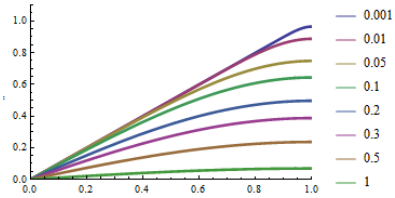
Пусть k=1. Четно – нечетное продолжение линейной функции $y=x$ с отрезка $[0,L]$ на всю ось имеет вид $\varphi(x)=(-1)^{[x/L]}stc(x,L)$, совпадающий с функцией $\varphi(x)$ из предыдущего примера.

```
φ[x_] = (-1)^Floor[x/2]stc[x, 2];
Plot[φ[x], {x, -2, 8}, AspectRatio → Automatic]
```



Эта функция имеет тот же вид, что и в предыдущем примере, только решение теперь надо рассматривать на отрезке [0,1], а не [0,2]. Имеем

```
a = 1;
u[x_, t_] := 1/√π NIntegrate[φ[x + 2a√t τ]Exp[-τ^2], {τ, -∞, ∞},
AccuracyGoal → 4, PrecisionGoal → 4, Method → "ClenshawCurtisRule"]
T = {0.001, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 1};
Plot[{u[x, 0.001], u[x, 0.01], u[x, 0.05], u[x, 0.1], u[x, 0.2], u[x, 0.3],
u[x, 0.5], u[x, 1]}, {x, 0, 1}, PlotRange → {{0, 1}, {0, 1.1}},
PlotStyle → {Thickness[0.01]}, PlotLegends → T]
```



Касательная к графику решения на правой границе в любой момент времени горизонтальна, т.е. $u'_x(L,t)=0$. Это означает, что тепловой поток равен нулю.

□

Пример. *Конечный стержень длины l. Начальная температура ноль. Температура левого конца равна* 1*, правого – ноль.* В соответствии с формулой (7) функция вида

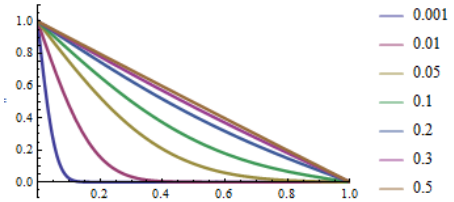
$$\tilde{u}(x,t)=\frac{1}{2l}\int_0^l\left(1-\frac{\xi}{l}\right)\left(\Theta_3\left(\pi\frac{\xi-x}{2l},e^{-\frac{\pi^2a^2t}{l^2}}\right)-\Theta_3\left(\pi\frac{\xi+x}{2l},e^{-\frac{\pi^2a^2t}{l^2}}\right)\right)d\xi$$

представляет решение уравнения теплопроводности с нулевыми граничными условиями и начальной температурой $u(x,0)=1-x/l$. Тогда функция $u(x,t)=1-\frac{x}{l}-\tilde{u}(x,t)$ будет представлять решение нашей задачи, поскольку она удовлетворяет уравнению теплопроводности, на левом конце принимает значение 1, на правом $x=l$ – принимает значение 0, и значение функции $u(x,t)$ при $t=0$ для всех $0<x<l$ будет 0. Т.о. общее решение нашей задачи будет иметь вид

$$u(x,t)=1-\frac{x}{l}-\frac{1}{2l}\int_0^l(1-\frac{\xi}{l})\left(\Theta_3(\pi\frac{\xi-x}{2l},e^{-\frac{\pi^2a^2t}{l^2}})-\Theta_3(\pi\frac{\xi+x}{2l},e^{-\frac{\pi^2a^2t}{l^2}})\right)d\xi$$

Построим графики распределения температуры в стержне для этой задачи.

```
l = 1; a = 1; φ[x_] := 1 - x/l;
θ3m[τ_, x_, t_] := EllipticTheta[3, π(τ - x)/(2l), e^(-π^2 a^2 t/l^2)]
θ3p[τ_, x_, t_] := EllipticTheta[3, π(τ + x)/(2l), e^(-π^2 a^2 t/l^2)]
u[x_, t_] := 1 - x/l - 1/(2l) NIntegrate[φ[τ](θ3m[τ, x, t] - θ3p[τ, x, t]),
                {τ, 0, l}, AccuracyGoal → 6, PrecisionGoal → 6]
T = {0.001, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5};
Plot[{u[x, 0.001], u[x, 0.01], u[x, 0.05], u[x, 0.1], u[x, 0.2],
    u[x, 0.3], u[x, 0.5]}, {x, 0, 1}, PlotRange → {{0, 1}, {-0.1, 1.1}},
    PlotStyle → {Thickness[0.01]}, PlotLegends → T]
```



В моменты времени $t\geq 0.5$ графики распределения температуры практически не изменяются и совпадают с последним графиком. Это значит, что в стержне установилась температура $u(x,t)=1-x$ и начиная с момента $t=0.5$ влияние начального условия пренебрежимо мало. □

Решение задачи распространения тепла в неограниченном во всех направлениях двумерном теле, имеющего в нулевой момент времени температуру $u(x,y,0)=\varphi(x,y)$, дается следующей формулой.

$$u(x,y,t)=\frac{1}{4\pi a^2 t}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\varphi(\xi,\eta)e^{-\frac{(x-\xi)^2+(y-\eta)^2}{4a^2t}}d\xi\,d\eta$$

Если начальная температура такова, что может быть представлена в виде $\varphi(x,y)=\varphi_1(x)\cdot\varphi_2(y)$, то этот интеграл преобразуется в произведение двух интегралов

$$u(x,y,t)=\frac{1}{4\pi a^2 t}\int_{-\infty}^{\infty}\varphi_1(\xi)e^{-\frac{(x-\xi)^2}{4a^2t}}d\,\xi\cdot\int_{-\infty}^{\infty}\varphi_2(\eta)e^{-\frac{(y-\eta)^2}{4a^2t}}d\,\eta$$

Если в них сделать замену $\tau=\frac{\xi-x}{2a\sqrt{t}}$ и $\nu=\frac{\eta-y}{2a\sqrt{t}}$, то мы получим

$$u(x,y,t)=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}\varphi_1\left(x+2a\sqrt{t}\,\tau\right)e^{-\tau^2}d\,\tau\times\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}\varphi_2\left(y+2a\sqrt{t}\,\nu\right)e^{-\nu^2}d\,\nu \qquad (8)$$

Пример. *Бесконечная плоскость. Начальная температура равна нулю везде, кроме области квадрата* $D=[-1,1]\times[-1,1]$, *в которой она равна единице*.

Поскольку $\varphi(x,y)=\varphi_1(x)\cdot\varphi_2(y)$, где $\varphi_i(u)=\begin{cases}1, & -1\le u\le 1\\ 0, & |u|>1\end{cases}$ $(i=1,2)$, то решение может быть представлено в форме (8). Имеем

```
φ[x_] = Piecewise[{{0, x ≤ -1}, {1, x < 1}}, 0];
a =.;
u[x_, t_] = FullSimplify[1/Sqrt[π] Integrate[φ[x + 2a Sqrt[t] τ] Exp[-τ^2],
                {τ, -∞, ∞}, Assumptions :→ t > 0 && a > 0]]
```

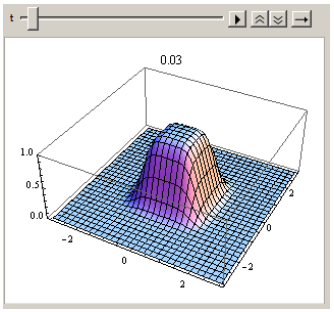
$$\frac{1}{2}\left(\mathrm{Erf}\left[\frac{1-x}{2a\sqrt{t}}\right]+\mathrm{Erf}\left[\frac{1+x}{2a\sqrt{t}}\right]\right)$$

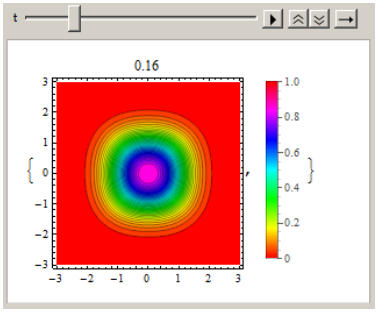
Теперь построим анимацию решения. Вначале представим решение как поверхность (панель анимации показана на следующем рисунке слева).

```
a = 1; U[x_, y_, t_] := u[x, t] * u[y, t];
T = Sequence[Join[{0.001}, Table[0.01 * i, {i, 80}]]];
Animate[Plot3D[Evaluate[U[x, y, t]], {x, -3, 3}, {y, -3, 3},
              PlotPoints → 31, PlotRange → {0, 1}, Mesh → Full,
              PlotLabel → t], {t, T}, AnimationRunning → False]
```

Во второй панели анимации представим решение контурным графиком (панель анимации показана на следующем рисунке справа).

```
Animate[{ContourPlot[U[x, y, t], {x, -3, 3}, {y, -3, 3}, Contours → 31,
              ColorFunction → Hue, ColorFunctionScaling → False,
              PlotPoints → 31, PlotRange → Full, PlotLabel → t],
             BarLegend[{(Hue[#]&), {0, 1}}]},
  {t, T}, AnimationRunning → False]
```





□

3.8.3 Одномерные колебания

Поперечные колебания струны, продольные колебания упругого стержня и многие другие явления описываются одномерным волновым уравнением

$$u''_{tt} = a^2 u''_{xx} + f(x,t) \tag{1}$$

Для однозначного определения решения необходимо задать начальные условия

$$u(x,0) = \varphi(x),\ u'_t(x,t) = \psi(x) \tag{2}$$

и граничные условия, если у колеблющегося объекта имеются границы.

Известно общее решение Даламбера задачи (1), (2) для неограниченной прямой [2]

$$u(x,t) = \frac{\varphi(x+at)+\varphi(x-at)}{2} + \frac{1}{2a}\int_{x-at}^{x+at}\psi(\alpha)d\alpha + \frac{1}{2a}\int_0^t \int_{x-a(t-\tau)}^{x+a(t-\tau)} f(\xi,\tau)d\xi\ d\tau \tag{3}$$

$(-\infty < x < \infty, t \ge 0)$. Формула (3) при любых функциях $\varphi(x), \psi(x)$ и $f(x,t)$ дает решение уравнения (1).

Основная идея построения интегрального решения первой краевой задачи для одномерного волнового уравнения (1) на конечном отрезке [*0,L*] при начальных условиях (2) и нулевых граничных условиях $u(0,t)=0,\ u(L,t)=0\quad t\ge 0$ состоит в следующем. В соответствии с методом продолжения выполняется нечетное периодическое продолжение начальных функций $\varphi(x), \psi(x)$ и $f(x,t)$ на всю ось Ox [8]. Подстановка продолженных таким образом функций в (3) дает функцию $u(x,t)$, являющуюся решением уравнения колебаний (1), удовлетворяющую на отрезке $[0, L]$ начальным условиям (2) и обращающуюся в ноль на концах этого отрезка. Известно [8, 9], что результирующая формула может быть представлена в виде

$$u(x,t) = \frac{1}{2}\left((-1)^{\left[\frac{x+at}{L}\right]}\varphi(stc(x+at,2L)) + (-1)^{\left[\frac{x-at}{L}\right]}\varphi(stc(x-at,2L)) \right) +$$

$$+\frac{1}{2a}\int_{stc(x-at,2L)}^{stc(x+at,2L)}\psi(\alpha)d\alpha + \frac{1}{2a}\int_0^t d\tau \int_{stc(x-a(t-\tau),2L)}^{stc(x+a(t-\tau),2L)} f(\xi,\tau)d\xi \quad (0\le x\le L, t\ge 0) \tag{4}$$

где квадратные скобки [*z*] используются для обозначения целой части числа *z*, а функция $stc(x,w)$ представляет периодическую пилообразную непрерывную кусочно-линейную функцию, определяемую формулой (2.6). Приведем ее здесь еще раз.

$$stx(x,w)=\left|x-w\left[\frac{x}{w}+\frac{1}{2}\right]\right| \quad \text{или} \quad stc(x,w)=\frac{w}{2\pi}\arccos\left(\cos\frac{2\pi x}{w}\right), \qquad (5)$$

В части 2 п.2.4.5 мы рассмотрели несколько примеров, в которых полагали начальную скорость равной нулю $u'_t(x,t)=\psi(x)=0$ и $f(x,t)=0$. Если положить начальное смещение и внешнее усилие равными нулю $\varphi(x)=0$ и $f(x,t)=0$, то решение примет вид

$$u(x,t)=\frac{1}{2a}\int_{stc(x-at,2L)}^{stc(x+at,2L)}\psi(\alpha)d\alpha \qquad (0\le x\le L, t\ge 0). \qquad (6)$$

Во всех примерах этого пункта будем полагать скорость распространения колебаний равной единице $a=1$.

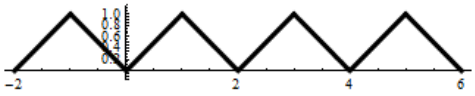
Пример. *Конечная струна длины L=1, закрепленная на концах. Начальное смещение ноль. Начальная скорость в форме ступеньки.* Струна возбуждается ударом жесткого плоского молоточка, сообщающего ей начальное распределение скоростей

$$\left.\frac{\partial u}{\partial t}\right|_{t=0}=\psi(x)=\begin{cases}0, & 0\le x<c-\delta\\ v_0, & c-\delta\le x\le c+\delta,\\ 0, & c+\delta<x\le L\end{cases}$$

где c – координата середины точки удара, δ – полуширина ступеньки.

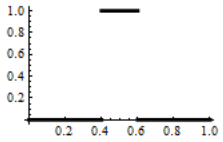
Пусть начальная скорость всюду ноль, кроме отрезка [0.4, 0.6], где она равна единице, т.е. $u'_t(x,0)=\psi(x)=\begin{cases}1, 0.4\le x\le 0.6\\ 0, x<0.4\vee x>0.6\end{cases}$. Решение будем строить, используя формулу (6). Определим функцию $stc(x,w)$.

```
stc[x_, w_] = Abs[x - w Floor[x/w + 1/2]];
Plot[stc[x, 2], {x, -2, 6}, AspectRatio → Automatic]
```



Создадим функцию начальной скорости $\psi(x)$.

```
ψ[x_] = Piecewise[{{0, x ≤ 0.4}, {1, x < 0.6}}, 0];
Plot[ψ[x], {x, 0, 1}]
```

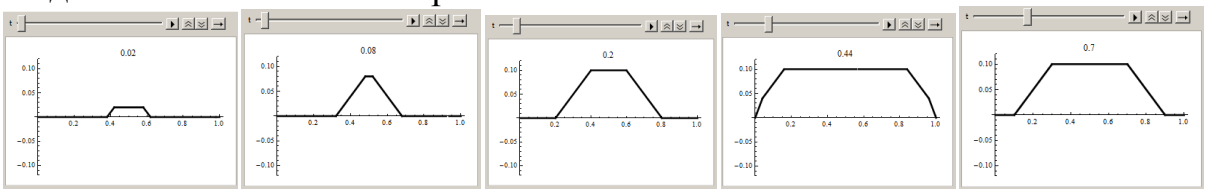


Создаем функцию решения и строим анимацию.

```
a =.; v[x_, t_] = 1/(2a) Simplify[Integrate[ψ[τ], {τ, α, β},
                    Assumptions :→ α > 0 && β > 0]]
u[x_, t_] = v[x, t]/.{α → stc[x - at, 2L], β-> stc[x + at, 2L]};
a = 1; L = 1; T = Range[0, 1.98, 0.02];
```

Animate[
Plot[$u[x,t]$, {x, **0**, **1**}, **PlotRange** → {−**0.12**, **0.12**}, **PlotLabel** → t],
{t, T}, **AnimationRunning** → **False**]

На следующем рисунке показаны профили струны в некоторые последовательные моменты времени.



Для этой краевой задачи в литературе приводится решение в виде ряда Фурье ([2], стр. 156-157)

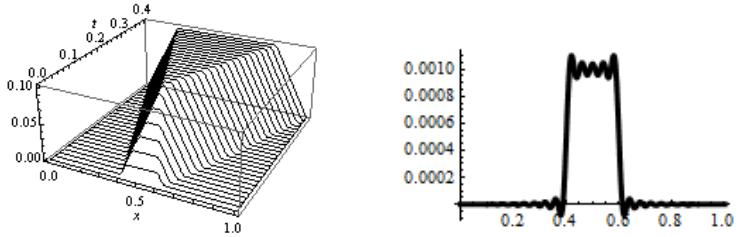
$$u(x,t)=\frac{4v_0 L}{\pi^2 a}\sum_{k=1}^{\infty}\frac{1}{k^2}\sin\frac{k\pi c}{L}\sin\frac{k\pi\delta}{L}\sin\frac{k\pi x}{L}\sin\frac{k\pi a t}{L}$$

Вод код для построения формы струны с помощью ряда.

$a = 1; L = 1; v0 = 1; \delta = 0.1; c = 0.5;$

$$uf[x\_, t\_] = \frac{4v0L}{\pi^2 a}\,Sum[\frac{1}{k^2}Sin[\frac{k\pi c}{L}]Sin[\frac{k\pi\delta}{L}]Sin[\frac{k\pi x}{L}]Sin[\frac{k\pi a t}{L}], \{k, 1, 50\}];$$

Plot3D[**uf**[x, t], {x, **0**, L}, {t, **0**, **0.4**}, **Mesh** → {**0**, **25**}, **PlotPoints** → **40**,
PlotStyle → **None**, **AxesLabel** → {x, t}]



На предыдущем рисунке слева представлены профили струны в дискретные моменты времени $0 \le t \le 0.4$ (период колебаний равен 2). Как видим, решения полученные обоими способами, совпадают. Но во втором решении мы ограничили сумму 50 – ю слагаемыми, и поэтому оно является приближенным. Если построить график решения в близкий к нулевому момент времени, то осцилляции, создаваемые частичной суммой ряда Фурье, будут заметны.

Plot[**uf**[x, **0.001**], {x, **0**, **1**}] (* предыдущий рисунок справа *)

Это связано с тем, что точное (обобщенное) решение принадлежит классу C^0, а не C^∞, которому принадлежит частичная сумма ряда Фурье.

□

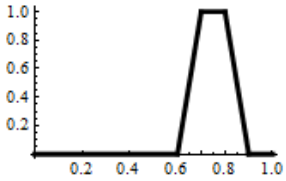
Пример. *Конечная струна, закрепленная на концах. Начальное смещение ноль. Внешнее усилие ноль. Начальная скорость в форме трапеции.*

Создаем функцию $\psi(x)$.

Clear[$\alpha, \beta, a, L, x, w, \psi, v, u$];

$$\psi[x\_] = \mathbf{10}\,Piecewise[\{\{0, x \le 0.6\}, \{x - 0.6, x < 0.7\}, \{0.1, x \le 0.8\}, \{0.9 - x, x \le 0.9\}\}, 0];$$

Plot[$\psi[x]$, {x, **0**, **1**}]



Поскольку функция `Integrate` умеет работать с кусочными функциями, то при построении решения будем использовать ее (а не `NIntegrate`).

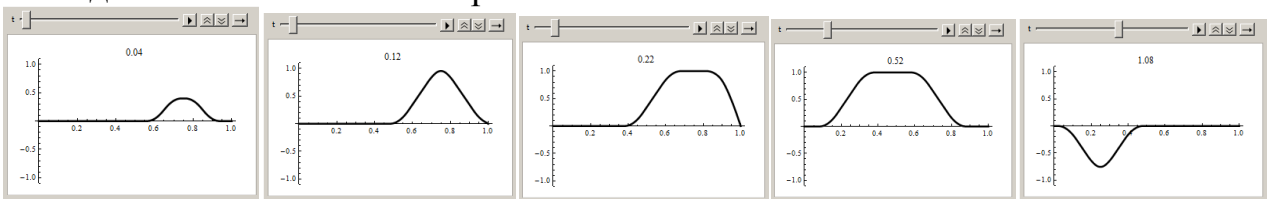
$$\boldsymbol{v[x\_, t\_] = \frac{1}{2a}\,Simplify[Integrate[\psi[\tau], \{\tau, \alpha, \beta\}, Assumptions :\to \alpha > 0 \;\&\&\; \beta > 0]]};$$

$$\boldsymbol{u[x\_, t\_] = v[x, t]/.\{\alpha \to stc[x - at, 2L], \beta -> stc[x + at, 2L]\}};$$

$$\boldsymbol{a = 1; L = 1; T = Range[0, 1.98, 0.02]};$$

$$\boldsymbol{Animate[Plot[u[x, t], \{x, 0, 1\}, PlotRange \to \{-0.12, 0.12\}, PlotLabel \to t], \{t, T\}, AnimationRunning \to False]}$$

На следующем рисунке показана панель анимации в некоторые последовательные моменты времени.



Здесь мы использовали код, создающий функцию `stc`, из предыдущего примера. Заметим, что при создании функции решения $u(x,t)$ можно было бы использовать функцию `NIntegrate`, например так

$$\boldsymbol{u[x\_, t\_] := (\alpha = stc[x - at, 2L]; \beta = stc[x + at, 2L]; \frac{1}{2a}\,NIntegrate[\psi[\tau], \{\tau, \alpha, \beta\}])};$$

Однако в этом случае вычисления будут выполняться значительно дольше.

□

При решении одномерного волнового уравнения на конечном отрезке часто используется метод разделения переменных. Рассмотрим, к каким формулам приводит этот метод при решении 1 – й краевой задачи для конечной струны.

Пусть $f(x,t)=0$, начальные условия имеют вид (2)

$$u(x,0)=\varphi(x),\ u'_t(x,t)=\psi(x)\ \ (0\le x\le L), \tag{2}$$

граничные условия равны 0 ($u(0,t)=u(L,t)=0$) и согласованы с начальными условиями, т.е. $\varphi(0)=\varphi(L)=0$. Разложим функции $\varphi(x), \psi(x)$ на отрезке [0,L] в ряд по синусам.

$$\varphi(x)=\sum_{n=1}^{\infty}\alpha_n \sin\frac{n\pi x}{L},\ \text{где}\ \alpha_n=\frac{2}{L}\int_0^L \varphi(x)\sin\frac{n\pi x}{L}\,dx$$

$$\psi(x)=\sum_{n=1}^{\infty}\beta_n \sin\frac{n\pi x}{L},\ \text{где}\ \beta_n=\frac{2}{L}\int_0^L \psi(x)\sin\frac{n\pi x}{L}\,dx$$

Тогда функция

$$u(x,t)=\sum_{n=1}^{\infty}\alpha_n \sin\frac{n\pi x}{L}\cos\frac{n\pi a t}{L}+\frac{\beta_n L}{n\pi a}\sin\frac{n\pi x}{L}\sin\frac{n\pi a t}{L} \qquad (7)$$

будет представлять решение уравнения (1), удовлетворять начальным условиям (2) и нулевым граничным условиям. То, что функция $u(x,t)$ удовлетворяет уравнению (1) и нулевым граничным условиям следует из того, что каждое слагаемое в (7) удовлетворяет уравнению и граничным условиям. Выполнение начальных условий проверяется подстановкой t=0 в выражение (7) и в ее производную по t. Действительно, из (7) при t=0 сразу получаем $u(x,0)=\sum_{n=1}^{\infty}\alpha_n \sin\frac{n\pi x}{L}=\varphi(x)$. Далее из (7) имеем

$$u'_t(x,0)=\sum_{n=1}^{\infty}\frac{\beta_n L}{n\pi a}\sin\frac{n\pi x}{L}\cos\frac{n\pi a t}{L}\left(\frac{n\pi a}{L}\right)\Bigg|_{t=0}=\sum_{n=1}^{\infty}\beta_n \sin\frac{n\pi x}{L}=\psi(x)$$

Таким образом, алгоритм решения уравнения колебаний конечной струны с закрепленными концами и нулевым внешним усилием состоит в следующем. Надо разложить начальные функции $\varphi(x),\psi(x)$ на отрезке [0,L] в ряд по синусам и, зная коэффициенты α_n, β_n построить ряд (7). Напишем общую процедуру решения уравнения колебаний конечной струны с закрепленными концами, реализующую формулу (7). Имеем

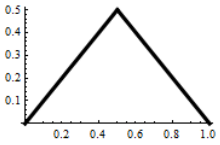
```
Clear[solFourier]
solFourier = Function[{x, t, a, L, φ, ψ, n}, Block[{fφ, fψ, gφ, gψ},
fφ = Chop[N[FourierSinSeries[φ[x], x, n,
                    FourierParameters → {1, π/L}]]];
fψ = Chop[N[FourierSinSeries[ψ[x], x, n,
                    FourierParameters → {1, π/L}]]];
gφ = fφ/.{Sin[w_] → Sin[w]Cos[w/x at]};
gψ = fψ/.{Sin[w_] → Sin[w]Sin[w/x at]/(w/x)};
gφ + gψ]];
```

Функция `solFourier` возвращает выражение частичной суммы (7), приближающей решение $u(x,t)$. Она принимает следующие аргументы: идентификаторы/имена переменных x и t, значение параметра a из уравнения (1), длину отрезка L, функции начального смещения φ и начальной скорости ψ (не выражения), количество членов `n` частичной суммы Фурье. Протестируем созданную функцию.

Пример. *Конечная струна, закрепленная на концах. Начальная скорость 0. Начальное смещение в форме треугольника.*

Создаем функцию смещения $\varphi(x)$.

```
φ[x_] = Piecewise[{{x, x ≤ 0.5}}, 1 - x];
Plot[φ[x], {x, 0, 1}]
```

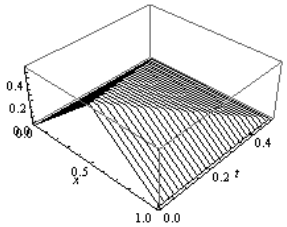


Создаем функцию решения $u(x,t)$ и строим график

$\boldsymbol{u[\mathrm{x\_},\mathrm{t\_}] = \mathbf{solFourier}[x,t,1,1,\varphi,0\&,50];}$

$\boldsymbol{\mathbf{Plot3D}[u[x,t],\{x,0,1\},\{t,0,0.5\},\mathbf{Mesh}\to\{0,25\},\mathbf{PlotPoints}\to 40,}$

$\boldsymbol{\mathbf{PlotStyle}{-}{>}\ \mathbf{None},\mathbf{AxesLabel}{-}{>}\ \{x,t\}]}$



Обратите внимание на то, как мы задаем нулевую скорость – в виде чистой функции 0&.

Сравните с решением этой задачи в п.2.4.5 (пример 5.4) предыдущей части. Решения идентичны, если не обращать внимание на то, что частичная сумма представляет приближенное решение. Выбор количества членов ряда равное 50 делает погрешность вычислений малой.

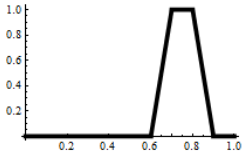
□

Пример. *Конечная струна, закрепленная на концах. Начальное смещение ноль. Начальная скорость в форме трапеции.*

$\boldsymbol{\psi[\mathrm{x\_}] = 10\mathbf{Piecewise}[\{\{0,x\le 0.6\},\{x-0.6,x<0.7\},\{0.1,x\le 0.8\},}$

$\boldsymbol{\{0.9-x,x\le 0.9\}\},0];}$

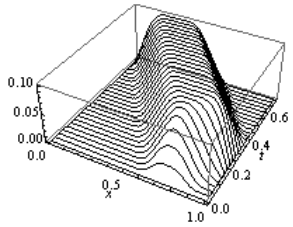
$\boldsymbol{\mathbf{Plot}[\psi[x],\{x,0,1\},\mathbf{PlotStyle}\to\{\mathbf{Black},\mathbf{Thickness}[0.02]\}]}$



$\boldsymbol{u[\mathrm{x\_},\mathrm{t\_}] = \mathbf{solFourier}[x,t,1,1,0\&,\psi,50];}$

$\boldsymbol{\mathbf{Plot3D}[u[x,t],\{x,0,1\},\{t,0,0.7\},\mathbf{Mesh}\to\{0,25\},\mathbf{PlotPoints}\to 40,}$

$\boldsymbol{\mathbf{PlotStyle}{-}{>}\ \mathbf{None},\mathbf{AxesLabel}{-}{>}\ \{x,t\}]}$



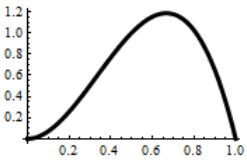
Сравните с решением этой задачи, полученной нами в этом пункте ранее. Решения идентичны.

□

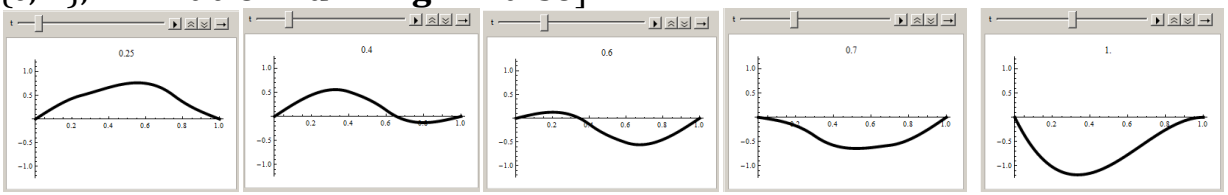
Пример. *Конечная струна, закрепленная на концах. Начальная скорость 0. Начальное смещение в форме кубической параболы.*

$\boldsymbol{\varphi[\mathrm{x\_}] = 8\,x^2(1-x);}$

$\boldsymbol{\mathbf{Plot}[\varphi[x],\{x,0,1\},\mathbf{PlotStyle}\to\{\mathbf{Black},\mathbf{Thickness}[0.02]\}]}$



```
L = 1; u[x_, t_] = solFourier[x, t, 1, L, φ, 0&,50];
T = Range[0, 1.975, .025];
Animate[Plot[u[x, t], {x, 0, L}, PlotStyle → {Black, Thickness[0.015]},
PlotRange → {−1.25, 1.25}, PlotLabel → t],
{t, T}, AnimationRunning → False]
```

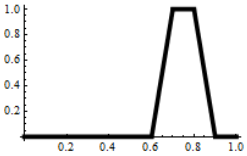


На рисунке показана панель анимации в моменты времени t = 0.25, 0.4, 0.6, 0.7, 1.0.

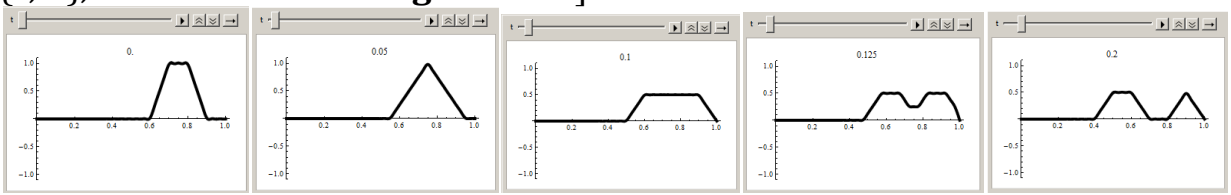
□

Пример. *Конечная струна, закрепленная на концах. Начальная скорость 0. Начальное смещение в форме трапеции.*

```
φ[x_] = 10Piecewise[{{0, x ≤ 0.6}, {x − 0.6, x < 0.7}, {0.1, x ≤ 0.8},
{0.9 − x, x ≤ 0.9}}, 0];
Plot[ψ[x], {x, 0, 1}, PlotStyle → {Black, Thickness[0.02]}]
```



```
L = 1; u[x_, t_] = solFourier[x, t, 1, L, φ, 0&, 50];
T = Range[0, 1.975, .025];
Animate[Plot[u[x, t], {x, 0, L}, PlotStyle → {Black, Thickness[0.015]},
        PlotRange → {−1.1, 1.1}, PlotLabel → t],
{t, T}, AnimationRunning → False]
```



На рисунке показана панель анимации в моменты времени t = 0, 0.05, 0.1, 0.125, 0.2. Обратите внимание на кадр в момент времени t=0. На графике видна некоторая рябь, что вызвано недостаточной точностью приближения решения частичной суммой с 50 – ю членами.

□

Выше мы рассмотрели несколько примеров, в которых полагали внешнее усилие равным нулю $f(x,t)=0$. Если положить начальное смещение и скорость равным нулю $\varphi(x)=0$ и $\psi(x)=0$, то для конечного отрезка решение (4) примет вид

$$u(x,t)=\frac{1}{2a}\int_0^t d\,\tau \int_{stc(x-a(t-\tau),2L)}^{stc(x+a(t-\tau),2L)} f(\xi,\tau)d\xi \quad (-\infty<x<\infty, t\ge 0) \tag{8}$$

Рассмотрим несколько примеров применения этой формулы.

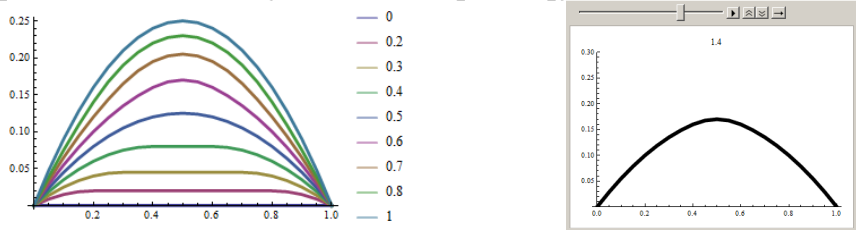
Пример. *Конечная струна, закрепленная на концах. Начальное смещение и скорость 0. Внешнее усилие постоянно.*

Струна находится в состоянии равновесия и в нулевой момент времени к ней прикладывается единичная сила, т.е. $\varphi(x)=0, \psi(x)=0, f(x,t)=1$. Начинается колебательный процесс. Решение может быть представлено по формуле (8). Внутренний интеграл вычисляется тривиально и задача сводится к однократному интегралу. Первообразную функции `stc` можно представить в символьном виде. Однако *Mathematica* это не умеет делать. Поэтому мы будем использовать численное интегрирование.

```
a = 1; L = 1;
u[x_, t_] := 1/(2a) NIntegrate[stc[x + a(t - τ), 2L] - stc[x - a(t - τ), 2L],
                               {τ, 0, t}, AccuracyGoal → 4, PrecisionGoal → 4]
```

Численное интегрирование кусочных функций выполняется довольно долго. Для ускорения работы мы вычислим значения функции $u(x,t)$ в дискретном наборе точек и построим график решения в виде ломаной с небольшим шагом между точками по оси x. Графики решения в моменты времени $t=0, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 1.0$ показаны на следующем рисунке слева.

```
T = Join[{0}, Range[0.2, 0.8, 0.1], {1}];
U = Table[{x, u[x, t]}, {t, T}, {x, 0, 1, 0.05}];
ListLinePlot[U, PlotStyle → {Thickness[0.01]}, PlotLegends → T]
```



Для построения анимации добавим следующий код.

```
T = Range[0, 2, 0.05];
U = Table[{x, u[x, t]}, {t, T}, {x, 0, 1, 0.05}];
tp = Table[ListLinePlot[U[[i]], PlotRange → {0., 0.3},
           PlotStyle → {Black, Thickness[0.015]}, PlotLabel → T[[i]]],
    {i, Length[T] - 1}];
ListAnimate[tp, AnimationRunning → False]
```

Один из кадров анимации показан на предыдущем рисунке справа.

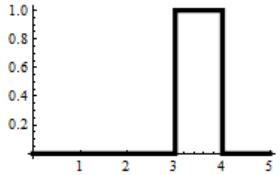
□

Пример. *Конечная струна, закрепленная на концах. Начальное смещение и скорость 0. Внешнее усилие не зависит от времени и имеет форму ступеньки.*

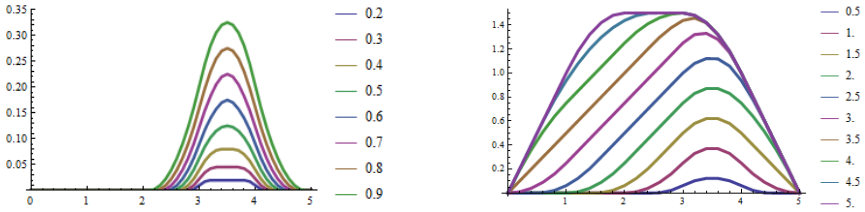
Струна длинной L=5 находится в состоянии равновесия $\varphi(x)=0, \psi(x)=0$ и в нулевой момент времени к ней прикладывается ступенчатая сила

$f(x,t)=\begin{cases}1, 3\le x\le 4\\ 0, x<3\vee x>4\end{cases}$ $\forall\, t>0$. Начинается колебательный процесс. Решение строим по формуле (8), используя численное интегрирование.

```
Clear[α, β]; a = 1; L = 5;
stc[x_, w_] = Abs[x − wFloor[x/w + 1/2]];
f[x_] = UnitBox[x − 3.5];
Plot[f[x], {x, 0, 5}]   (* график внешнего усилия *)
```



```
F[α_, β_] = Integrate[f[t], {t, α, β},
                Assumptions :→ α >= 0 && β >= 0 && α < β]
G[x_, t_, τ_] = F[stc[x − a(t − τ), 2L], stc[x + a(t − τ), 2L]];
u[x_, t_] := 1/(2a) NIntegrate[G[x, t, τ], {τ, 0, t},
                AccuracyGoal → 4, PrecisionGoal → 4];
T1 = Range[0.2, 0.9, 0.1];
U1 = Table[{x, u[x, t]}, {t, T1}, {x, 0, L, 0.1}];
ListLinePlot[U1, PlotStyle → {Thickness[0.01]},
PlotLegends → T1, PlotRange → {0, 0.35}]   (* левый график на след. рис. *)
```



```
T2 = Range[0.5, 5, 0.5];
U2 = Table[{x, u[x, t]}, {t, T2}, {x, 0, L, 0.2}];
ListLinePlot[U2, PlotStyle → {Thickness[0.01]},
        PlotLegends → T2]   (* правый график на предыдущем рисунке *)
```

На левом графике показаны профили струны в начальные моменты времени t = 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9; на правом – в моменты времени t = 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0.

□

3.8.4 Примеры решения краевых задач для уравнение Лапласа

Много прикладных задач сводится к решению уравнения Лапласа $\Delta u = 0$. Для однозначного определения решения к уравнению следует добавить граничные условия и условия убывания на бесконечности, если область неограничена. К уравнению Лапласа приводятся задачи исследования движения идеальной жидкости, электростатики, задачи стационарной теплопроводности, статического прогиба мембран и многие другие. Функции, удовлетворяющие уравнению Лапласа называются гармоническими.

Решение уравнения Лапласа является сложной задачей и явных формульных решений известно очень мало. Здесь мы рассмотрим несколько примеров для которых решение уравнения Лапласа можно представить в интегральной форме.

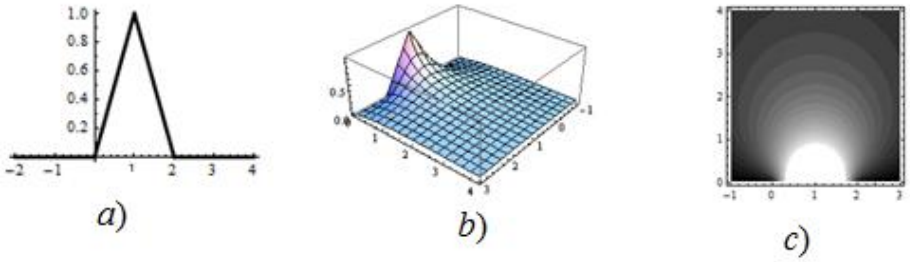
Гармоническая функция в верхней полуплоскости определяется по формуле

$$u(x,y)=\frac{1}{\pi}\int_{-\infty}^{\infty}u(\xi)\frac{y}{(x-\xi)^2+y^2}d\xi, \qquad (1)$$

где $u(\xi)$ граничное значение функции. Если это граничное значение финитное (имеет конечный носитель – интервал на котором функция отлична от нуля), то интеграл можно вычислять по этому интервалу. Формулу (1) нельзя использовать при значении y=0, т.е. на границе области.

Пример. *Гармоническая функция в верхней полуплоскости. Граничное значение в форме треугольника*.

```
φ[x_] = Piecewise[{{0, x < 0}, {x, x < 1}, {2 - x, x < 2}}, 0];
Plot[φ[x], {x, -2, 4}]   (* следующий рисунок а *)
u[x_, y_] := 1/π (NIntegrate[ξ y/((x - ξ)^2 + y^2), {ξ, 0, 1}, WorkingPrecision → 5] +
                  NIntegrate[(2 - ξ) y/((x - ξ)^2 + y^2), {ξ, 1, 2}, WorkingPrecision → 5]);
Plot3D[u[x, y], {x, -1, 3}, {y, 0.01, 4}, PlotRange → All, PlotPoints → 40]
ContourPlot[u[x, y], {x, -1, 3}, {y, 0.01, 4}, ContourLines → False,
            PlotPoints → 25, Contours → 20, ColorFunction → GrayLevel]
```



a) граничная функция $u(\xi)$; *b*) график/поверхность функции $u(x,y)$;
c) контурный график функции $u(x,y)$.

□

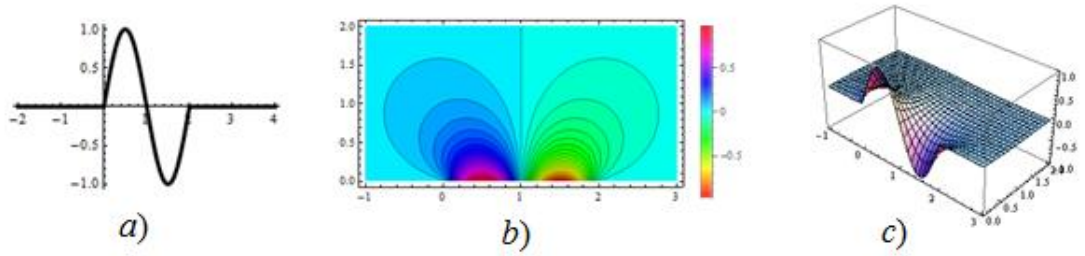
Пример. *Гармоническая функция в верхней полуплоскости. Граничное значение в форме одной волны синусоиды*.

```
φ[x_] = Piecewise[{{0, x < 0}, {Sin[πx], x < 2}}, 0];
Plot[φ[x], {x, -2, 4}]
u[x_, y_] := 1/π NIntegrate[φ[ξ] y/((x - ξ)^2 + y^2), {ξ, 0, 2}, MaxRecursion → 100,
   PrecisionGoal → 4, AccuracyGoal → 4, Method -> "TrapezoidalRule"]
```

Поскольку численное интегрирование нашей функции выполняется довольно долго, то для ускорения работы мы вычислим значения функции $u(x,y)$ в дискретном наборе точек, который мы выбираем, а не в наборе точек, который

выбирают сами графические функции. Для построения графиков решения по дискретному набору точек в пакете *Mathematica* имеются подходящие функции.

```
X = Range[-1, 3, 0.05];
Y = Join[{0.01}, Range[0.05, 2, 0.05]];
W = Flatten[Table[{x, y, u[x, y]}, {y, Y}, {x, X}], 1];
ListContourPlot[W, Contours → 40, ColorFunction → Hue,
  PlotRange → All, AspectRatio → Automatic, PlotLegends → Automatic]
ListPlot3D[W, PlotRange → All, BoxRatios → {4, 2, 2}, Mesh → {41, 16, 0}]
```



a) граничная функция $u(x,0)$; *b*) контурный график функции $u(x,y)$; *c*) график/поверхность функции $u(x,y)$.

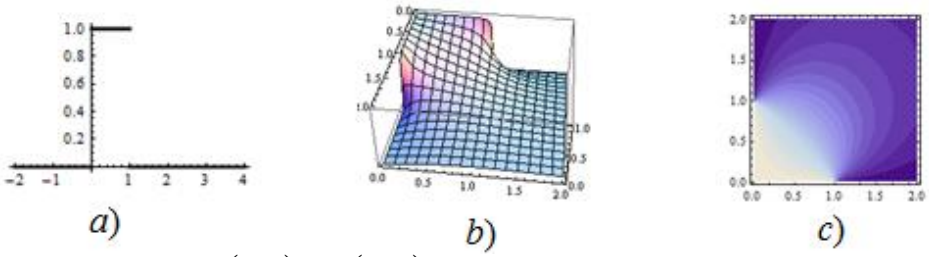
□

Используя метод продолжения, можно получить решение уравнения Лапласа в первом квадранте (в области $x \ge 0,\ y \ge 0$). Пусть граничные значения в 1-ом квадранте имеют вид $u(x,0) = \varphi(x),\ x \ge 0$, $u(0,y) = \psi(y),\ y \ge 0$. Решение уравнения Лапласа в 1-ом квадранте имеет вид

$$u(x,y) = \frac{4xy}{\pi}\int_0^\infty \left(\frac{\varphi(\xi)\xi}{((x-\xi)^2+y^2)((x+\xi)^2+y^2)} + \frac{\psi(\xi)\xi}{((y-\xi)^2+x^2)((y+\xi)^2+x^2)} \right) d\xi$$

Пример. *Гармоническая функция в первом квадранте. Граничное значение в форме прямоугольных столбиков на обеих полуосях X и Y.*

```
φ[x_] = Piecewise[{{0, x < 0}, {1, x < 1}}, 0];
Plot[φ[x], {x, -2, 4}]
u[x_, y_] := 4xy/π (NIntegrate[ξ φ[ξ]/((x - ξ)^2 + y^2)((x + ξ)^2 + y^2),
    {ξ, 0, 1}, MaxRecursion → 12, AccuracyGoal → 4,
    PrecisionGoal → 4] +
  NIntegrate[ξ φ[ξ]/((y - ξ)^2 + x^2)((y + ξ)^2 + x^2), {ξ, 0, 1},
    MaxRecursion → 12, AccuracyGoal → 4,
    PrecisionGoal → 4])
Plot3D[u[x, y], {x, 0.01, 2}, {y, 0.01, 2}, PlotRange → All, PlotPoints → 40]
ContourPlot[u[x, y], {x, 0.01, 2}, {y, 0.01, 2}, ContourLines → False,
  PlotPoints → 25, Contours → 20]
```



a) граничная функция $u(\tau,0)=u(0,\tau)$; *b*) график/поверхность функции $u(x,y)$; *c*) контурный график функции $u(x,y)$.

□

Пример. *Гармоническая функция в первом квадранте. Граничное значение в форме треугольников на обеих полуосях X и Y.*

$\boldsymbol{\varphi}[\mathrm{x\_}] = \mathbf{Piecewise}[\{\{\mathbf{0}, x < 0\}, \{x, x < 1\}, \{\mathbf{2} - x, x < 2\}\}, \mathbf{0}];$

$\mathbf{Plot}[\varphi[x], \{x, -\mathbf{2}, \mathbf{4}\}]$

$$\boldsymbol{u}[\mathrm{x\_}, \mathrm{y\_}] := \frac{4xy}{\pi}\left(\mathbf{NIntegrate}\left[\frac{\xi\varphi[\xi]}{((x-\xi)^2+y^2)((x+\xi)^2+y^2)}, \{\xi, 0, 2\}, \mathbf{AccuracyGoal} \to 4, \mathbf{PrecisionGoal} \to 4, \mathbf{MaxRecursion} \to 12\right] + \mathbf{NIntegrate}\left[\frac{\xi\varphi[\xi]}{((y-\xi)^2+x^2)((y+\xi)^2+x^2)}, \{\xi, 0, 2\}, \mathbf{AccuracyGoal} \to 4, \mathbf{PrecisionGoal} \to 4, \mathbf{MaxRecursion} \to 12\right]\right)$$

Ускорим построение графиков вычислением значений функции в некотором наборе точек.

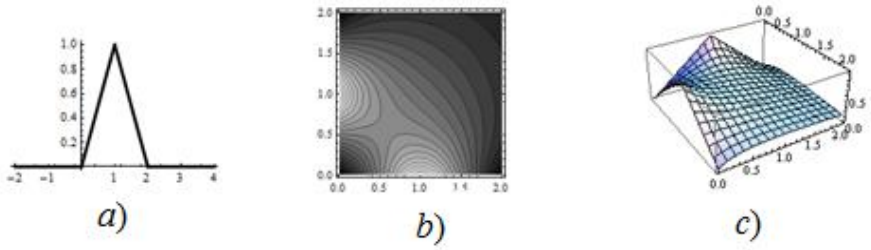
$X = \mathbf{Join}[\{\mathbf{0.01}\}, \mathbf{Range}[\mathbf{0.05}, \mathbf{2}, \mathbf{0.05}]];$

$Y = \mathbf{Join}[\{\mathbf{0.01}\}, \mathbf{Range}[\mathbf{0.05}, \mathbf{2}, \mathbf{0.05}]];$

$W = \mathbf{Flatten}[\mathbf{Table}[\{x, y, u[x, y]\}, \{y, Y\}, \{x, X\}], \mathbf{1}];$

$\mathbf{ListContourPlot}[W, \mathbf{Contours} \to \mathbf{20}, \mathbf{ColorFunction} \to \mathbf{GrayLevel}]$

$\mathbf{ListPlot3D}[W]$



a) граничная функция $u(\tau,0)=u(0,\tau)$; *b*) контурный график функции $u(x,y)$; *c*) график/поверхность функции $u(x,y)$.

□

Гармоническая в круге радиуса R функция вычисляется по формуле Пуассона

$$u(r,\varphi)=\frac{1}{2\pi}\int_0^{2\pi} f(\theta)\frac{R^2-r^2}{R^2-2Rr\cos(\theta-\varphi)+r^2}\,d\theta \qquad (2)$$

где $f(\theta)$ граничное значение функции, а r, φ - полярные координаты точки внутри круга.

Пример. *Стационарное распределение температуры внутри бесконечного кругового цилиндра. Температура на поверхности имеет вид косинусоиды.*

Ось цилиндра направим вдоль оси Oz. Температура точек u внутри цилиндра не зависит от z и будет удовлетворять двумерному уравнению Лапласа внутри круга радиуса R. Если использовать полярные координаты, то граничное условие для функции температуры $u(r,\varphi)$ будет иметь вид $u(R,\theta)=f(\theta)$. Сама функция $u(r,\varphi)$ будет определяться формулой (2).

Пусть $f(\theta)=\cos 3\theta$. Тогда

$$u(r,\varphi)=\frac{R^2-r^2}{2\pi}\int_0^{2\pi}\frac{\cos(3\theta)}{R^2-2Rr\cos(\theta-\varphi)+r^2}\,d\theta$$

Сделаем замену $\theta-\varphi=t$ и, учитывая периодичность (не нужно менять пределы интегрирования), получим

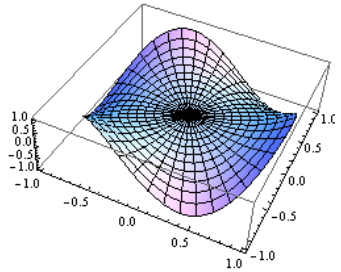
$$u(r,\varphi)=\frac{R^2-r^2}{2\pi}\int_0^{2\pi}\frac{\cos(3t+3\varphi)}{R^2-2Rr\cos t+r^2}\,dt$$

Тогда

```
R =.;
u[r_, φ_] = Simplify[(R^2 - r^2)/(2 π) Integrate[Cos[3 t + 3 φ]/(R^2 - 2 R r Cos[t] + r^2), {t, 0, 2 π},
                Assumptions :> (0 < r < R && φ ∈ Reals && R > 0)]]
```

$$\frac{r^3\cos[3\varphi]}{R^3}$$

```
R = 1;
RevolutionPlot3D[u[r, φ], {r, 0, 1}, {φ, 0, 2 π},
                Mesh → {11, 41, 0}, PlotRange → All]
```



Замечание. Если замену $\theta-\varphi=t$ не делать, то *Mathematica* 9 возвращает неверный результат! Попробуйте.

```
Simplify[(R^2 - r^2)/(2 π) Integrate[Cos[3 t]/(R^2 - 2 R r Cos[t - φ] + r^2), {t, 0, 2 π},
                Assumptions :> (0 < r < R && φ ∈ Reals && R > 0)]]
```

$$\frac{(r^6-R^6)\cos[3\varphi]}{2r^3R^3}$$

□

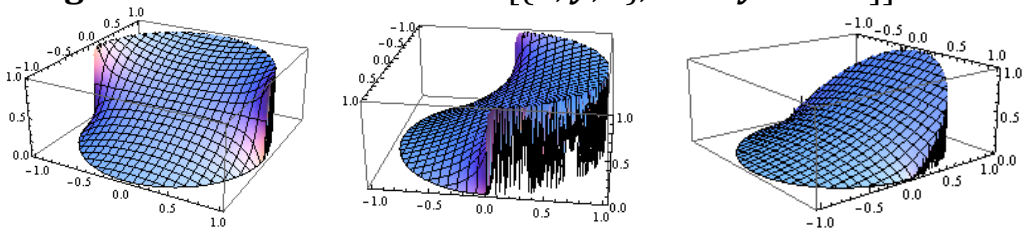
Пример. *Стационарное распределение температуры внутри бесконечного кругового цилиндра. Температура на поверхности кусочно постоянна.*

Пусть на поверхности бесконечного кругового цилиндра радиуса R поддерживается следующая температура: $u=T$ при $0<\varphi<\pi$; $u=0$ при $\pi<\varphi<2\pi$; (для любых *z* и *t*). Внутри цилиндра температура установилась и ее требуется найти. Задача сводится к решению двумерного уравнения Лапласа $\Delta u=0$ с граничным условием $u(R,\theta)=f(\theta)=\begin{cases}T, 0<\theta<\pi\\0, \pi<\theta<2\pi\end{cases}$. Решение дается интегралом Пуассона (2). Подставляя в него $f(\theta)=0$ при $\pi<\theta<2\pi$ и T при $0<\theta<\pi$, получим

$$u(x,y)=T\frac{R^2-x^2-y^2}{2\pi}\int_0^{\pi}\frac{1}{R^2-2R(x\cos\theta+y\sin\theta)+x^2+y^2}d\theta$$

Обратите внимание на замену $r\cos(\theta-\varphi)=x\cos\theta+y\sin\theta$, которую мы сделали в интеграле. Имеем

```
R = 1; T = 1;
u[x_, y_] := T (R^2 - x^2 - y^2)/(2 π) *
                NIntegrate[(1/(R^2 - 2 R (x Cos[θ] + y Sin[θ]) + x^2 + y^2)),
                  {θ, 0, π}, AccuracyGoal → 4, PrecisionGoal → 4];
U[x_, y_] := If[x^2 + y^2 < R^2, u[x, y], 0];
Plot3D[U[x, y], {x, -R, R}, {y, -R, R}, PlotRange → {0, T},
       PlotPoints → 40, Mesh → 21,
       RegionFunction → Function[{x, y, z}, x^2 + y^2 < R^2]]
```



Если покрутить график, то вы заметите «рябь» у контура круга (рисунок в середине). Это вызвано тем, что интеграл Пуассона дает решение краевой задачи *внутри* круга, а граничное значение $f(\theta)$ он только приближает. Использовать его для вычисления значения функции $u(r,\varphi)$ на границе нельзя. В точках границы погрешность численного интегрирования вызывает «рябь». Даже если граничная функция $f(\theta)$ будет непрерывной, то «рябь» остается. Пусть, например, граничное значение имеет вид $f(\theta)=\begin{cases}\sin\theta, 0\le\theta<\pi\\0, \pi\le\theta<2\pi\end{cases}$. Тогда для построения графика решения в предыдущем коде надо изменить строку, определяющую функцию $u(x,y)$

```
u[x_, y_] := (R^2 - x^2 - y^2)/(2π) NIntegrate[
                Sin[θ]/(R^2 - 2R(xCos[θ] + ySin[θ]) + x^2 + y^2), {θ, 0, π},
                AccuracyGoal → 4, PrecisionGoal → 4];
```

График решения показан на предыдущем рисунке справа..

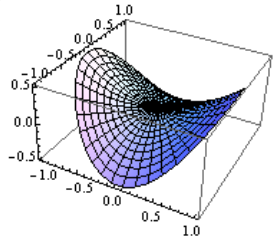
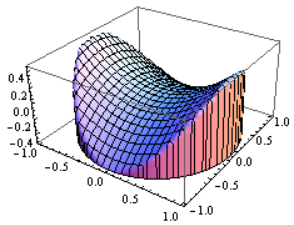
□

Пример. *Тонкая мембрана натянута на проволочный каркас,* проектирующийся на плоскость Oxy в окружность радиуса R с центром в начале координат. Уравнение контура мембраны в цилиндрических координатах имеет вид: $u(R,\theta)=f(\theta)=h\cos 2\theta$ $(0\le\theta<2\pi)$. Известно, что форма поверхности $z=u(x,y)$, по которой расположится пленка, удовлетворяет уравнению Лапласа и граничному условию $u(R,\theta)=f(\theta)$. Найдем эту форму, используя интеграл Пуассона (2). Прежде, чем вычислять интеграл, сделаем в нем замену $\theta-\varphi=t$ и учтем периодичность подынтегральной функции (не нужно менять пределы интегрирования). Имеем

```
R =.; h =.;
u[r_, φ_] = Simplify[h (R^2 - r^2)/(2π)
              Integrate[Cos[2t + 2φ]/(R^2 - 2Rr Cos[t] + r^2), {t, 0, 2π},
              Assumptions :→ (0 < r < R && φ ∈ Reals && R > 0)]]
```

$$\frac{h\,r^2\cos[2\varphi]}{R^2}$$

```
R = 1; h = 0.5;
RevolutionPlot3D[u[r, φ], {r, 0, 1}, {φ, 0, 2π}, Mesh → {11, 41, 0},
    PlotRange → All, BoxRatios → {2R, 2R, 2h}] (* след. рисунок слева *)
```

Делая замену $x=r\cos\varphi$, $y=r\sin\varphi$, переходим к представлению формы мембраны в декартовых координатах.

```
U[x_, y_] = h (x^2 - y^2)/R^2;
RegionPlot3D[z < U[x, y] && x^2 + y^2 < R^2,
  {x, -R, R}, {y, -R, R}, {z, -h, h}, PlotRange → All, PlotPoints → 41,
  Mesh → {21, 21, 0}, BoxRatios → {2R, 2R, 2h}] (* пред. рисунок справа *)
```

Литература.

1. Очан Ю.С. Методы математической физики.
2. Н.С. Кошляков, Э.Б.Глинер, М.М.Смирнов Основные дифференциальные уравнения математической физики. ГИФМЛ.М. 1962, 767с.
3. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
4. Э.Т.Уиттекер, Дж.Н.Ватсон Курс современного анализа. Ч.2. ГИФМЛ. М. 1963г. 515с.
5. В.М. Бабич, М.Б. Капилевич, С.Г. Михлин и др. Под. ред. С.Г. Михлина. Справочная математическая библиотека. Линейные уравнения математической физики. – М.: Наука, 1964. – 368 с.
6. Полянин А.Д. Справочник по линейным уравнениям математической физики. М.: ФИЗМАТЛИТ, 2001. – 576с.
7. Карслоу Г., Егер Д. Теплопроводность твердых тел. – М.: Наука, 1964.– 487 с.
8. Доля П.Г. Периодическое продолжение функций и решение уравнения колебаний струны в системах символьной математики.// Вестник Харьк. нац. ун-та., - 2006.- № 733. Сер. ”Математическое моделирование.
9. Dolya P.G. Solution of the initial value problem for the inhomogeneous equation of vibrations of a finite string with homogeneous boundary conditions // Kharkiv: Verkin Institute for Low Temperature Physics and Engineering of NASU. Lapunov memorial conference., Book of abstracts, – p. 38-39, 2007