



# Система компьютерной верстки $\text{\LaTeX}$

---

Фадеев Е.П.

2022

МГУ • Физический факультет

# Введение

---

*TeX — система компьютерной вёрстки, разработанная американским профессором информатики Дональдом Кнутом в целях создания компьютерной типографии. В неё входят средства для секционирования документов, для работы с перекрёстными ссылками. В частности, благодаря этим возможностям, TeX популярен в академических кругах, особенно среди математиков и физиков.*

[Wikipedia](#)

# TeX vs. MS Word

TeX	MS Word
What you see is what you mean	What you see is what you get
Созданный документ выглядит одинаково на всех устройствах	Созданный документ может выглядеть иначе на другом устройстве
Удобно набирать формулы	Неудобно набирать формулы
Сравнительно легко организовать автоматическую нумерацию глав, разделов, формул, таблиц, иллюстраций и т.п., а также перекрестные ссылки на них	Сравнительно трудоемко организовать автоматическую нумерацию глав, разделов, формул, таблиц, иллюстраций и т.п., а также перекрестные ссылки на них
Содержимое документа и его форматирование изолированы друг от друга	Содержимое документа и его форматирование сложно изолировать друг от друга

# T<sub>E</sub>X vs. MS Word

T <sub>E</sub> X	MS Word
What you see is what you mean	What you see is what you get
Просмотр текста и его набор — разные операции	Текст одновременно набирается и просматривается
Трудно набирать страницы со сложным макетом	Макет страницы можно настраивать на лету

# $\text{\TeX}$ из командной строки

Для работы с  $\text{\TeX}$  необходимо установить дистрибутив, например, **MiK $\text{\TeX}$**  или **TeXLive**.

После установки дистрибутива исходный текст можно скомпилировать в “**.dvi**” командой

```
$ latex helloworld.tex
```

**helloworld.tex** →  $\text{\LaTeX}$  → **helloworld.dvi**

А в “**.pdf**” командой

```
$ pdflatex helloworld.tex
```

**helloworld.tex** → pdf $\text{\LaTeX}$  → **helloworld.pdf**

## Редактор $\text{\LaTeX}$

При этом набирать исходный текст можно почти в любом текстовом редакторе. Однако гораздо удобнее установить специальный редактор под  $\text{\LaTeX}$ .

Автор предпочтает редактор **TeXstudio**, который поддерживает проверку и подсветку синтаксиса, автоматическое дополнение команд, выделение ошибок, предварительный просмотр и многое другое. Если установить дистрибутив  $\text{\LaTeX}$  до установки *TeXstudio*, то редактор сам найдет дистрибутив и настроит трансляцию исходного кода документа в **pdf** файл.

### Предупреждение

По умолчанию *TeXstudio* ничего не знает про русский язык и будет подсвечивать каждое слово написанное на кириллице, как грамматическую ошибку. Чтобы это исправить, необходимо установить русский словарь. Инструкция доступна по ссылке.

# Простейший документ

Простейший документ выглядит примерно так.

```
\documentclass{article}
\begin{document}
Hello world, \LaTeX{}.
\end{document}
```

## Документ и преамбула

- Все, что идёт после `\documentclass{article}` но до `\begin{document}`, называется преамбулой документа.
- Все, что идет после `\begin{document}` но до `\end{document}`, задает содержимое документа.
- Все, что идет после символа `%`, считается комментарием и игнорируется  $\text{\LaTeX}$  до конца этой строки.

```
\documentclass{article}
% preamble
% комментарий в преамбуле
\begin{document}
% content
% комментарий в документе
\end{document}
```

## Преамбула

В преамбуле задаётся стиль документа и подключаются пакеты. Например, для того, чтобы использовать кириллицу в  $\text{\LaTeX}$ , необходимо подключить пару пакетов.

```
\documentclass{article}

\usepackage[T2A]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[russian]{babel}

\begin{document}
    Привет мир, \LaTeX{}.
\end{document}
```

## Преамбула

Ещё пара пакетов имеет смысл импортировать по-умолчанию, если вы планируете набирать формулы.

```
\documentclass{article}

\usepackage[T2A]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[russian]{babel}

\usepackage{amsmath}
\usepackage{amssymb}

\begin{document}
    Привет мир, \LaTeX{}.
\end{document}
```

## Набор текста

Слова разделяются пробелами,  
а абзацы ---  
пустыми строками.

Абзацный отступ в исходном  
тексте оставлять  
не  
надо: он получается  
автоматически.

Слова разделяются пробелами, а  
абзацы — пустыми строками.

Абзацный отступ в исходном тек-  
сте оставлять не надо: он получается ав-  
томатически.

## Специальные символы

{	Начало группы
}	Конец группы
\$	Начало или конец формулы
#	Аргумент при определении пользовательской команды
&	Отделение элементов матрицы или таблицы
%	Комментарий
-	Нижний индекс в формуле
^	Верхний индекс в формуле
~	Неразрывный пробел
\	Начало команды

Таблица 1: Специальные символы

## Простейшие команды

Команды начинаются с символа “\”. Самые простейшие команды позволяют напечатать некоторые специальные символы в их исходном виде.

Курс тугрика повысился на 7\%,  
и теперь за него дают \\$200.

Курс тугрика повысился на 7%, и теперь  
за него дают \$200.

Остальные команды состоят из символов латинского алфавита. Например, команда \TeX печатает символ ТЕХ. Ещё, например, есть команды меняющие шрифт текста.

Этот текст обычного размера, \Large а этот более крупного.

Этот текст обычного размера, а этот более крупного.

## Простейшие команды

Команды начинаются с символа “\”. Самые простейшие команды позволяют напечатать некоторые специальные символы в их исходном виде.

Курс тугрика повысился на 7\%,  
и теперь за него дают \\$200.

Курс тугрика повысился на 7%, и теперь  
за него дают \$200.

Остальные команды состоят из символов латинского алфавита. Например, команда \TeX печатает символ ТЕХ. Ещё, например, есть команды меняющие шрифт текста.

Полужирный шрифт начнется с \bfseries этого слова. Снова \mdseries светлый, теперь \slshape наклонный, до нового переключения; вновь \upshape прямой..

Полужирный шрифт начнется с этого слова. Снова светлый, теперь наклонный, до нового переключения; вновь прямой..

## Простейшие команды

Команды начинаются с символа “\”. Самые простейшие команды позволяют напечатать некоторые специальные символы в их исходном виде.

Курс тугрика повысился на 7\%,  
и теперь за него дают \\$200.

Курс тугрика повысился на 7%, и теперь  
за него дают \$200.

Остальные команды состоят из символов латинского алфавита. Например, команда \TeX печатает символ ТЕХ. Ещё, например, есть команды меняющие шрифт текста.

Полужирный шрифт начнется с \bfseries этого слова. Снова \mdseries светлый, теперь \slshape наклонный, до нового переключения; вновь \upshape прямой..

Полужирный шрифт начнется с этого слова. Снова светлый, теперь наклонный, до нового переключения; вновь прямой.

# Группы

Группы обозначаются парой фигурных скобок. Некоторые команды действуют в группах локально.

Полужирным шрифтом набрано только `{\bfseries это}` слово; после скобок все идет, как прежде.

Полужирным шрифтом набрано только **это** слово; после скобок все идет, как прежде.

Как правило, по выходу из группы модифицирующие команды сбрасываются.

Сначала `{переключим шрифт на \itshape курсив; теперь сделаем шрифт еще и {\bfseries полужирным;}` посмотрите, как восстановится} шрифт после кон`{ца г}руппы.`

Сначала переключим шрифт на курсив; теперь сделаем шрифт еще и полужирным; посмотрите, как восстановится шрифт после конца группы.

## Команды с аргументами

Обязательные аргументы указываются у команды в фигурных скобках, необязательные в квадратных.

```
\documentclass[12pt,twocolumn]{book}
```

Здесь

- `\documentclass` — команда;
- `book` — обязательный аргумент;
- `12pt` и `twocolumn` — необязательные аргументы;

# Окружения

Ещё одна важная конструкция в  $\text{\LaTeX}$  — окружения (environment).

Окружение начинается с команды `\begin{имя_окружения}`, а заканчивается командой `\end{имя_окружения}`. Например, все содержимое документа задаётся в окружении `document`.

```
\begin{center}
```

Все строки этого абзаца будут центрированы; переносов не будет, если только какое-то слово, как в дезоксирибонуклеиновой кислоте, не длинней строки.

```
\end{center}
```

Все строки этого абзаца будут центрированы; переносов не будет, если только какое-то слово, как в дезоксирибонуклеиновой кислоте, не длинней строки.

## Автоматическая генерация ссылок

Команда `\label` создаёт метку, на которую потом можно создать перекрестную ссылку. Команда `\pageref` подставит номер страницы, на которой располагается указанная метка.

```
\textbf{Теорема Пифагора.}\label{pythagoras} В прямоугольном  
треугольнике квадрат гипотенузы равен сумме квадратов катетов.
```

Из теоремы Пифагора (см. с. [\pageref{pythagoras}](#)) получаем, что длина гипотенузы равна  $\sqrt{3^2 + 4^2} = 5$ .

**Теорема Пифагора.** В прямоугольном треугольнике квадрат гипотенузы равен сумме квадратов катетов.

Из теоремы Пифагора (см. с. 18) получаем, что длина гипотенузы равна  $\sqrt{3^2 + 4^2} = 5$ .

## Набор формул

---

## Включенные и выключенные формулы

Формулы бывают внутри текста («включенные») и выделенные в отдельную строку («выключенные»).

- Включенные формулы набираются между одинарных знаков **\$** или между пары скобок вида **\(\backslash\)**.

Оказалось, что **\$y\$** вдвое больше, чем **\(x\)**: **\$y=2x\$**.

Оказалось, что  $y$  вдвое больше, чем  $x$ :  $y = 2x$ .

- Выключенные формулы набираются, например, между двойных знаков **\$\$**.

Оказалось, что **\$y\$** вдвое больше, чем **\$x\$**: **\$\$y=2x.**

Оказалось, что  $y$  вдвое больше, чем  $x$ :

$$y = 2x.$$

## Выключенные формулы

Также выключенные формулы можно набирать с помощью скобок `\[ \]`.

Оказалось, что  $y$  вдвое больше, чем  $x$ :  
$$y = 2x.$$

Оказалось, что  $y$  вдвое больше, чем  $x$ :  
$$y = 2x.$$

---

А также окружений `equation` и `equation*`.

Оказалось, что  $y$  вдвое больше, чем  $x$ :

```
\begin{equation*}  
y=2x.  
\end{equation*}
```

Оказалось, что  $y$  вдвое больше, чем  $x$ :  
$$y = 2x.$$

## Автоматическая нумерация выключенных формул

Окружение `equation` без звездочки автоматически нумерует формулу, что позволяет делать перекрестные ссылки, если поставить внутри формулы метку.

Оказалось, что  $\$y\$$  вдвое больше, чем  $\$x\$$ :

```
\begin{equation}\label{eq}
    y=2x.
\end{equation}
```

Если  $\$x=0.5\$$ , то по формуле `\eqref{eq}` со страницы `\pageref{eq}` находим, что  $\$y=1\$$ .

Оказалось, что  $y$  вдвое больше, чем  $x$ :

$$y = 2x. \tag{1}$$

Если  $x = 0.5$ , то по формуле (1) со страницы 21 находим, что  $y = 1$ .

## Индексы

Нижние индексы в формулах указываются после символа “\_”.

$$\$x\_1\$ \qquad \qquad x_1$$

---

$$\$x\_i+1\$ \qquad \qquad x_{i+1}$$

---

$$\$x_{\{i+1\}}\$ \qquad \qquad x_{i+1}$$

---

$$\$x_{\{i\_j\}}\$ \qquad \qquad x_{i_j}$$

## Степени

Степени или верхние индексы в формулах указываются после символа “ $\wedge$ ”.

$$\$x^2\$ \qquad \qquad x^2$$

---

$$\$x^{2n}\$ \qquad \qquad x^{2n}$$

---

$$\$x^{\{2n\}}\$ \qquad \qquad x^{2n}$$

---

$$\$x^{\{n^2\}}\$ \qquad \qquad x^{n^2}$$

## Степени и индексы. Примеры

Катеты  $a$ ,  $b$  треугольника связаны с его гипотенузой  $c$  формулой  $c^2 = a^2 + b^2$  (см. с. [\pageref{pythagoras}](#)).

Катеты  $a$ ,  $b$  треугольника связаны с его гипотенузой  $c$  формулой  $c^2 = a^2 + b^2$  (см. с. [18](#)).

Из теоремы Ферма следует, что уравнение  $x^{4357} + y^{4357} = z^{4357}$  не имеет решений в натуральных числах.

Из теоремы Ферма следует, что уравнение

$$x^{4357} + y^{4357} = z^{4357}$$

не имеет решений в натуральных числах.

## Степени и индексы. Примеры

Катеты  $a$ ,  $b$  треугольника связаны с его гипотенузой  $c$  формулой  $c^2 = a^2 + b^2$  (см. с. [\pageref{pythagoras}](#)).

Катеты  $a$ ,  $b$  треугольника связаны с его гипотенузой  $c$  формулой  $c^2 = a^2 + b^2$  (см. с. [18](#)).

Из теоремы Ферма следует, что уравнение  $x^{4357} + y^{4357} = z^{4357}$  не имеет решений в натуральных числах.

Из теоремы Ферма следует, что уравнение

$$x^{4357} + y^{4357} = z^{4357}$$

не имеет решений в натуральных числах.

## Дроби

Самая простая дробь записывается с помощью знака деления “/”.

Решая уравнение  $2x=1$ ,  
получаем  $x=1/2$ .

Решая уравнение  $2x = 1$ , получаем  $x = 1/2$ .

Для записи дробей используются команды `\frac` имеющая два обязательных параметра: числитель и знаменатель дроби. `\frac` сжимает дробь, чтобы она не вылезала за пределы строки, её аналог `\dfrac` нет, но это может привести к увеличению междустрочного интервала.

Решая уравнение  $2x=1$ ,  
получаем  $x=\frac{1}{2}$ .

Решая уравнение  $2x = 1$ , получаем  $x = \frac{1}{2}$ .

Решая уравнение  $2x=1$ ,  
получаем  $x=\frac{1}{2}$ .

Решая уравнение  $2x = 1$ , получаем  $x = \frac{1}{2}$ .

## Корни

Корни набираются с помощью команды `\sqrt`. Обязательный аргумент — подкоренное выражение, необязательный аргумент — показатель корня.

По общепринятым соглашениям,  $\sqrt[3]{x^3} = x$ ,  
но  $\sqrt{x^2} = |x|$ .

По общепринятым соглашениям,  $\sqrt[3]{x^3} = x$ , но  $\sqrt{x^2} = |x|$ .

## Штрихи

Штрихи (например, для обозначения производной) вводятся символом одинарной кавычки “'”.

Согласно формуле Лейбница,

$$\begin{aligned} \text{\textbackslash [} \\ (\mathbf{f}\mathbf{g})' = & \mathbf{f}'\mathbf{g} + 2\mathbf{f}'\mathbf{g}' + \mathbf{f}\mathbf{g}''. \end{aligned}$$

\]

Это похоже на формулу квадрата суммы.

Согласно формуле Лейбница,

$$(\mathbf{f}\mathbf{g})'' = \mathbf{f}''\mathbf{g} + 2\mathbf{f}'\mathbf{g}' + \mathbf{f}\mathbf{g}''.$$

Это похоже на формулу квадрата суммы.

Команда `\prime` печатает такой штрих: '/. Постановка одинарной кавычки в формуле эквивалента возведению в степень `\prime`.

`$x'`

$x'$

---

`$x^\prime$`

$x'$

## Многоточия

Команда `\cdots` печатает символы по центру строки ( $\cdots$ ), а команда `\ldots` снизу (...).

В детстве К.-Ф. Гаусс придумал,  
как быстро найти сумму

```
\[  
1+2+\cdots+100=5050;  
\]
```

это случилось, когда школьный  
учитель задал классу найти  
сумму чисел \$1, 2, \ldots, 100\$.

В детстве К.-Ф. Гаусс придумал,  
как быстро найти сумму

$$1 + 2 + \cdots + 100 = 5050;$$

это случилось, когда школьный  
учитель задал классу найти сум-  
му чисел  $1, 2, \dots, 100$ .

## Общее для формул



Формула  $y = 2x$   
--- тоже самое, что и  
формула  $y=2x$ .

Формула  $y = 2x$  — тоже самое, что и формула  $y = 2x$ .

- Можно расставлять пробелы разного размера в формулах вручную командами `\quad`, `\quadquad`, `\,,\,:`, `\; , \!`.
  - В выключенных формулах  $\text{\LaTeX}$  игнорирует все переносы строк. В включенных формулах допускается один перенос подряд.

## Символы греческого алфавита

Для каждого символа греческого алфавита есть своя команда.

$\alpha$	<code>\alpha</code>	$\beta$	<code>\beta</code>	$\gamma$	<code>\gamma</code>	$\delta$	<code>\delta</code>	$\epsilon$	<code>\epsilon</code>
$\zeta$	<code>\zeta</code>	$\eta$	<code>\eta</code>	$\theta$	<code>\theta</code>	$\iota$	<code>\iota</code>	$\kappa$	<code>\kappa</code>
$\lambda$	<code>\lambda</code>	$\mu$	<code>\mu</code>	$\nu$	<code>\nu</code>	$\xi$	<code>\xi</code>	$\pi$	<code>\pi</code>
$\rho$	<code>\rho</code>	$\sigma$	<code>\sigma</code>	$\tau$	<code>\tau</code>	$\upsilon$	<code>\upsilon</code>	$\phi$	<code>\phi</code>
$\chi$	<code>\chi</code>	$\psi$	<code>\psi</code>						

Таблица 2: Строчные символы греческого алфавита

### Замечание

Эти команды работают только внутри формул.

## Символы греческого алфавита

Пусть  $\alpha$ ,  $\beta$  и  $\gamma$  — углы треугольника, тогда

$$\begin{aligned} & \text{\textbackslash [} \\ & \alpha + \beta + \gamma = \pi. \\ & \text{\textbackslash ]} \end{aligned}$$

Пусть  $\alpha$ ,  $\beta$  и  $\gamma$  — углы треугольника, тогда

$$\alpha + \beta + \gamma = \pi.$$

## Символы греческого алфавита

Некоторые символы имеют несколько вариантов написания.

$\phi$	<code>\phi</code>	$\varphi$	<code>\varphi</code>
$\epsilon$	<code>\epsilon</code>	$\varepsilon$	<code>\varepsilon</code>
$\kappa$	<code>\kappa</code>	$\varkappa$	<code>\varkappa</code>
$\rho$	<code>\rho</code>	$\varrho$	<code>\varrho</code>
$\pi$	<code>\pi</code>	$\varpi$	<code>\varpi</code>
$\sigma$	<code>\sigma</code>	$\varsigma$	<code>\varsigma</code>
$\theta$	<code>\theta</code>	$\vartheta$	<code>\vartheta</code>

Таблица 3: Строчные символы греческого алфавита

## Символы греческого алфавита

Команды для прописных символов греческого алфавита начинаются с прописных букв, если их начертание не совпадает с прописными символами латинского алфавита.

$\Gamma$	<code>\Gamma</code>	$\Delta$	<code>\Delta</code>	$\Theta$	<code>\Theta</code>	$\Lambda$	<code>\Lambda</code>	$\Xi$	<code>\Xi</code>
$\Pi$	<code>\Pi</code>	$\Sigma$	<code>\Sigma</code>	$\Upsilon$	<code>\Upsilon</code>	$\Phi$	<code>\Phi</code>	$\Psi$	<code>\Psi</code>
$\Omega$	<code>\Omega</code>								

Таблица 4: Прописные символы греческого алфавита

## Символы греческого алфавита

Начертание прописного символа  $\alpha$  совпадает с заглавной буквой латинского алфавита  $A$ . Символа  $\sigma$  в прописном варианте имеет вид  $\Sigma$  и не имеет аналога в латинском алфавите.

В  $\text{\TeX{}}$  символы  $\Sigma$  и  $\Pi$  не используются для обозначения операторов суммирования и произведения.

Начертание прописного символа  $\alpha$  совпадает с заглавной буквой латинского алфавита  $A$ . Символа  $\sigma$  в прописном варианте имеет вид  $\Sigma$  и не имеет аналога в латинском алфавите.

В  $\text{\TeX}$  символы  $\Sigma$  и  $\Pi$  не используются для обозначения операторов суммирования и произведения.

## Бинарные операторы, операторы сравнения и стрелки

+	+	-	-	±	\pm	⊤	\mp
*	*	×	\times	·	\cdot	○	\circ
÷	\div	:	\colon	\setminus	\setminus	/	/
∪	\cup	∩	\cap	∨	\vee	∧	\wedge
=	=	≠	\neq	≡	\equiv	≐	\doteq
~	\sim	≈	\approx	∝	\propto	≅	\cong
»	\gg	«	\ll	<	<	>	>
≤	\leq	≥	\geq	⩾	\geqslant	⩷	\leqslant
⊥	\perp	∈	\in	∉	\notin	∋	\ni
⊂	\subset	⊆	\subseteq	⊃	\supset	⊇	\supseteq

Таблица 5: Бинарные операторы и операторы сравнения

## Бинарные операторы, операторы сравнения и стрелки

$\rightarrow$	<code>\to</code>	$\longrightarrow$	<code>\longrightarrow</code>	$\Rightarrow$	<code>\Rrightarrow</code>	$\Longrightarrow$	<code>\Longrightarrow</code>
$\leftarrow$	<code>\leftarrow</code>	$\longleftarrow$	<code>\longleftarrow</code>	$\Leftarrow$	<code>\leftarrow</code>	$\Longleftarrow$	<code>\Longleftarrow</code>

Таблица 6: Стрелки

$\text{\LaTeX}$  автоматически добавляет пробелы слева и справа от бинарных операторов, операторов сравнения и стрелок.

$$\$x \textcolor{blue}{\backslash Leftrightarrow} y \$ \qquad \qquad x \Leftrightarrow y$$

---

$$\$x () y \$ \qquad \qquad x () y$$

## Бинарные операторы, операторы сравнения и стрелки

В англоязычной литературе принято использовать  $\leq$  и  $\geq$  (`\le` и `\ge`), а в русскоязычной  $\leqslant$  и  $\geqslant$  (`\leqslant` и `\geqslant`). Т.к. Дональд Кнут ориентировался в первую очередь на соотечественников, то англоязычные варианты этих знаков имеют более короткие команды.

Командой `\renewcommand{\le}{\leqslant}` в преамбуле файла можно подменить  $\leq$  на  $\leqslant$  во всем документе.

## Бинарные операторы, операторы сравнения и стрелки

\mid тоже считается оператором. Его часто употребляют при определении множеств.

`$M=\{x\in A\mid x>0\}$`

$$M = \{x \in A \mid x > 0\}$$

---

`$M=\{x\in A \mid x>0\}$`

$$M = \{x \in A | x > 0\}$$

Символ “:” интерпретируется как оператор деления. При записи отображений необходимо использовать команду “\colon” вместо “:”.

`$\colon X\rightarrow Y$`

$$f : X \rightarrow Y$$

---

`$f: X\rightarrow Y$`

$$f : X \rightarrow Y$$

## Операторы без пределов

log	<code>\log</code>	lg	<code>\lg</code>	ln	<code>\ln</code>	exp	<code>\exp</code>
sin	<code>\sin</code>	arcsin	<code>\arcsin</code>	cos	<code>\cos</code>	arccos	<code>\arccos</code>
tan	<code>\tan</code>	arctan	<code>\arctan</code>	cot	<code>\cot</code>		
tg	<code>\tg</code>	arctg	<code>\arctg</code>	ctg	<code>\ctg</code>	arcctg	<code>\arcctg</code>
sinh	<code>\sinh</code>	cosh	<code>\cosh</code>	tanh	<code>\tanh</code>	coth	<code>\coth</code>
sh	<code>\sh</code>	ch	<code>\ch</code>	th	<code>\th</code>	cth	<code>\cth</code>
arg	<code>\arg</code>	dim	<code>\dim</code>	sec	<code>\sec</code>	csc	<code>\csc</code>

**Таблица 7:** Функции. Команды выделенных строк доступны только при подключении пакета “babel”

## Операторы без пределов

Все операторы с предыдущего слайда несут свойство `\nolimits`. Это значит, что если у них поставить индексы, то они появятся в обычном месте, как нижний индекс или как степень.

$$\log_{10} x = \frac{\ln x}{\ln 10}$$

$$\exp^{-1} x = \ln x$$

Ещё эти операторы ставят небольшой пробел между функцией и её аргументом.

Сравните `\sin x` и `\mathrm{sin}x`. Сравните  $\sin x$  и  $\mathrm{sin}x$ .

Команда `\operatorname{operatorname}` позволяет создать функцию для одноразового применения.

$$\operatorname{arccot} x.$$

## Операторы с пределами

$\sum$	<code>\sum</code>	$\prod$	<code>\prod</code>	$\bigcup$	<code>\bigcup</code>	$\bigcap$	<code>\bigcap</code>
$\min$	<code>\min</code>	$\inf$	<code>\inf</code>	$\max$	<code>\max</code>	$\sup$	<code>\sup</code>
$\lim$	<code>\lim</code>	$\varliminf$	<code>\varliminf</code>	$\varlimsup$	<code>\varlimsup</code>		

Таблица 8: Операторы с пределами

```
\[
\sum_{k=1}^{\infty} \frac{1}{k^2}
= \frac{\pi^2}{6}
]
```

Формулы внутри текста, такие как  
`$\sum_{k=1}^{\infty}\frac{1}{k^2}`  
`= \frac{\pi^2}{6}$,`  
занимают меньше места по высоте.

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$$

Формулы внутри текста, такие как  $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$ , занимают меньше места по высоте.

## Операторы с пределами

Можно все равно заставить выводить индексы в качестве пределов, явно указав пределы после команды `\limits`.

Сумма первых  $n$  натуральных чисел выражается формулой

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}.$$

Сумма первых  $n$  натуральных чисел выражается формулой  $\sum_{i=1}^n i = \frac{n(n+1)}{2}$ .

# Операторы с пределами

$\int$	<code>\int</code>	$\iint$	<code>\iint</code>	$\iiint$	<code>\iiint</code>	$\iiiiint$	<code>\iiiiint</code>
$\oint$	<code>\oint</code>	$\int \cdots \int$	<code>\idotsint</code>				

Таблица 9: Знаки интегралов

У знака интеграла пределы появляются сбоку вне зависимости от режима.

```
\[  
\int_a^b f(x)\,dx  
\]
```

$$\int_a^b f(x) dx$$

## Замечание

“`\,`” необходимо в формуле, чтобы поставить правильный пробел между  $f(x)$  и  $dx$ .

## Разные символы

$\partial$	<code>\partial</code>	$'$	<code>\prime</code>	$\nabla$	<code>\nabla</code>	$\infty$	<code>\infty</code>
$\emptyset$	<code>\emptyset</code>	$\varnothing$	<code>\varnothing</code>	$\forall$	<code>\forall</code>	$\exists$	<code>\exists</code>
$\hbar$	<code>\hbar</code>	$\imath$	<code>\imath</code>	$\jmath$	<code>\jmath</code>	$'$	<code>\prime</code>

Таблица 10: Разные символы

```
\[
\frac{\partial f(x,y)}{\partial x} =
\lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x}
]
```

$$\frac{\partial f(x, y)}{\partial x} = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x}$$

## Скобки

(	(	)	)	[	[	]	]
{	{	}	\}				\
<	\langle	>	\rangle	\backslash	\backslashbackslash	/	/
[	\lfloor	]	\rfloor	\lceil	\lceil	\rceil	\rceil

Таблица 11: Скобки

```
\[  
\|\vec{x}-\vec{y}\|_2^2 = \sum_{i=1}^n (x_i - y_i)^2  
\]
```

$$\|\vec{x}-\vec{y}\|_2^2 = \sum_{i=1}^n (x_i - y_i)^2$$

```
\[  
\|\vec{x}-\vec{y}\|_1^2 = \sum_{i=1}^n |x_i - y_i|  
\]
```

$$\|\vec{x}-\vec{y}\|_1^2 = \sum_{i=1}^n |x_i - y_i|$$

## Скобки

Команды `\left` и `\right` позволяют автоматически подбирать размер скобки под габариты содержимого между ними.

```
\[  
e = \lim_{n\rightarrow\infty}(1 + \frac{1}{n})^n  
\]
```

$$e = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$$

```
\[  
e = \lim_{n\rightarrow\infty}\left(1 + \frac{1}{n}\right)^n  
\]
```

$$e = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$$

Эти команды всегда должны идти парой!

## Скобки

Знак деления является скобкой неслучайно: иногда хочется увеличивать его.

```
\[M(f)=  
\left(\int\limits_a^bf(x)\,dx\right)  
/(b-a)\]
```

$$M(f) = \left( \int_a^b f(x) dx \right) / (b - a)$$

Используя точку вместо левой скобки, можно указать  $\text{\LaTeX}$ , под размер чего подгонять правую скобку.

```
\[  
M(f)=\left.  
\left(\int\limits_a^bf(x)\,dx\right)  
\right/(b-a)  
\]
```

$$M(f) = \left( \int_a^b f(x) dx \right) / (b - a)$$

## Скобки

```
\[
\int\limits_a^b x^{-2} dx = -\left.\frac{1}{x}\right|_a^b
\]
```

$$\int_a^b x^{-2} dx = -\frac{1}{x} \Big|_a^b$$

## Скобки

Иногда автоматически подобранный размер неудачен. В таких случаях можно вручную указать конкретный размер скобки.

(	\bigl(	(	\Bigl(	(	\biggl(	(	\Biggl(
)	\bigr)	)	\Bigr)	)	\biggr)	)	\Biggr)

Таблица 12: Увеличенные кобки

```
$$\left| |x+1| - |x-1| \right|$$
```

$$||x + 1| - |x - 1||$$

```
$$\bigl| |x+1| - |x-1| \bigr|$$
```

$$||x + 1| - |x - 1||$$

## Диакритические знаки

$\hat{a}$	<code>\hat{a}</code>	$\check{a}$	<code>\check{a}</code>	$\tilde{a}$	<code>\tilde{a}</code>	$\acute{a}$	<code>\acute{a}</code>	$\grave{a}$	<code>\grave{a}</code>
$\dot{a}$	<code>\dot{a}</code>	$\ddot{a}$	<code>\ddot{a}</code>	$\breve{a}$	<code>\breve{a}</code>	$\bar{a}$	<code>\bar{a}</code>	$\vec{a}$	<code>\vec{a}</code>

Таблица 13: Диакритические знаки

```
\[
\ddot{a} \eqdef \frac{d^2 a}{dt^2}
\]
```

$$\ddot{a} \stackrel{\text{def}}{=} \frac{d^2 a}{dt^2}$$

```
\[
\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i
\]
```

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

## Диакритические знаки

Диакритические знаки с прошлого слайда имеют фиксированную ширину.

$$\$ \hat{f*g} = \hat{f} \cdot \hat{g} \$$$
$$f \hat{*} g = \hat{f} \cdot \hat{g}$$
$$\$ \vec{AC} = \vec{AB} + \vec{BC} \$$$
$$\vec{AC} = \vec{AB} + \vec{BC}$$

В таких ситуациях лучше использовать команды `\widehat`, `\overline`, `\overrightarrow` и т.п.

$$\$ \widehat{f*g} = \hat{f} \cdot \hat{g} \$$$
$$\widehat{f * g} = \hat{f} \cdot \hat{g}$$

```
\[  
\overrightarrow{AC} =  
\overrightarrow{AB} +  
\overrightarrow{BC}  
\]
```

$$\overrightarrow{AC} = \overrightarrow{AB} + \overrightarrow{BC}$$

# Матрицы

Матрицы с круглыми скобками набираются с помощью окружения `pmatrix`.

Матрицы задаются по строкам, строки отделяются с помощью `\backslash`, а столбцы с помощью `&`.

```
\[  
\begin{pmatrix}  
    a_1 & a_2  
\end{pmatrix}  
\]
```

$$\begin{pmatrix} a_1 & a_2 \end{pmatrix}$$

```
\[  
\begin{pmatrix}  
    a_1 \\ a_2  
\end{pmatrix}  
\]
```

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$$

## Матрицы: pmatrix

```
\[
\begin{pmatrix}
a_{11} - \lambda & a_{12} & a_{13} \\
a_{21} & a_{22} - \lambda & a_{23} \\
a_{31} & a_{32} & a_{33} - \lambda
\end{pmatrix}
```

$$\begin{pmatrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{pmatrix}$$

## Матрицы: matrix

```
\[
\begin{matrix}
a_{11} - \lambda & a_{12} & a_{13} \\
a_{21} & a_{22} - \lambda & a_{23} \\
a_{31} & a_{32} & a_{33} - \lambda
\end{matrix}
\]
```

$$\begin{matrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{matrix}$$

## Матрицы: vmatrix

```
\[
\begin{vmatrix}
a_{11} - \lambda & a_{12} & a_{13} \\
a_{21} & a_{22} - \lambda & a_{23} \\
a_{31} & a_{32} & a_{33} - \lambda
\end{vmatrix}
```

$$\begin{vmatrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{vmatrix}$$

## Матрицы: Vmatrix

```
\[
\begin{Vmatrix}
a_{11} - \lambda & a_{12} & a_{13} \\
a_{21} & a_{22} - \lambda & a_{23} \\
a_{31} & a_{32} & a_{33} - \lambda
\end{Vmatrix}
\]
```

$$\begin{vmatrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{vmatrix}$$

## Матрицы: bmatrix

```
\[
\begin{bmatrix}
a_{11} - \lambda & a_{12} & a_{13} \\
a_{21} & a_{22} - \lambda & a_{23} \\
a_{31} & a_{32} & a_{33} - \lambda
\end{bmatrix}
```

$$\begin{bmatrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{bmatrix}$$

## Матрицы: `bmatrix`

Матрицы входят в формулы как обычные символы крупного размера.

```
\[
\vec{a} = \begin{bmatrix}
a_x \\
a_y \\
a_z
\end{bmatrix}\]
```

$$\vec{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

## Матрицы: `bmatrix`

Многоточия в матрице можно поставить командами `\cdots`, `\vdots` и `\ddots`.

```
\[  
\begin{vmatrix}  
 1 & \cdots & 1 \\  
 \vdots & \ddots & \vdots \\  
 1 & \cdots & 1  
\end{vmatrix} = 0  
\]
```

$$\begin{vmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{vmatrix} = 0$$

## Пользовательские команды

---

## Определение пользовательских функций

В  $\text{\LaTeX}$  нет встроенных команд для принятых в России сокращений ряда функций. Например, действительная часть комплексного числа обозначается зарубежом  $\Re(z)$  ( $\text{\textcolor{blue}{\text{\textsf{Re}}}}(\text{\textcolor{red}{z}})$ ), а комплексная часть  $\Im(z)$  ( $\text{\textcolor{blue}{\text{\textsf{Im}}}}(\text{\textcolor{red}{z}})$ ). В России же приняты обозначения  $\text{Re } z$  и  $\text{Im } z$ , но соответствующих команд в  $\text{\LaTeX}$  нет.

```
$\text{\textcolor{red}{Re}} \textcolor{green}{z}$
```

$\text{Re} z$

Не должно быть наклона.  $\text{\textsf{mathsf}}$  позволяет писать прямым шрифтом.

```
$\text{\textcolor{red}{\textsf{Re}}} \textcolor{green}{z}$
```

$\text{Re} z$

Между функцией и её аргументом должен быть небольшой пробел.  $\text{\textsf{mathop}}$  решает эту проблему.

```
$\text{\textcolor{red}{\textsf{mathop}}}\{\text{\textcolor{blue}{\textsf{Re}}}\} \textcolor{green}{z}$
```

$\text{Re } z$

## Определение пользовательских функций

Если в тексте часто встречается взятие действительной части комплексного числа, то писать каждый раз `\mathop{\mathsf{Re}}` становится утомительно и чревато ошибками. Это самая простая ситуация, когда полезно определить команду для пользовательской функции командой `\DeclareMathOperator{cmd}{text}` в преамбуле документа.

```
% preamble
\DeclareMathOperator{\re}{Re}
\DeclareMathOperator{\im}{Im}
```

Указав две строки выше в преамбуле, можно в тексте использовать команды `\re` и `\im`.

```
$z = \re z + i \im z$
```

$$z = \operatorname{Re} z + i \operatorname{Im} z$$

## Определение пользовательских команд

Команда `\newcommand` позволяет определять пользовательские команды-макросы. Например, рассмотрим символ  $\stackrel{\text{def}}{=}$ , который получается с помощью выражения `\stackrel{\mathrm{def}}{=}`.

```
$x^2\stackrel{\mathrm{def}}{=} x \cdot x$
```

$$x^2 \stackrel{\text{def}}{=} x \cdot x$$

Следующей командой в преамбуле документа определяется новая команда `\eqdef`, которая эквивалента сочетанию `\stackrel{\mathrm{def}}{=}`:

```
% preamble  
\newcommand{\eqdef}{\stackrel{\mathrm{def}}{=}}
```

Это сокращает запись предыдущей формулы.

```
$x^2\eqdef x \cdot x$
```

$$x^2 \stackrel{\text{def}}{=} x \cdot x$$

## Определение пользовательских команд

Так, например, команда `\DeclareMathOperator` на самом деле под капотом работает превращается в `\newcommand` с небольшими изменениями в аргумент. Например, следующее определение для действительной части комплексного числа

```
\DeclareMathOperator{\re}{Re}
```

в точности соответствует

```
\newcommand{\re}{\mathop{\mathrm{Re}}}\nolimits
```

Здесь `\nolimits` влияет на то, как будут отображаться индексы, что в данном случае не принципиально.

<code>\nolimits</code>	<code>\limits</code>
$\mathrm{Re}_i^j$	$\mathrm{Re}_i^j$

## Определение пользовательских команд

Предположим, что в тексте часто встречается действительная ось  $\mathbb{R}$ , которая получается командой `\mathbb{R}` и мы хотим определить команду `\R` для этого.

Такая команда будет работать только внутри формулы.

```
\newcommand{\R}{\mathbb{R}}
```

Такая — только снаружи.

```
\newcommand{\R}{$\mathbb{R}$}
```

А такая — везде.

```
\newcommand{\R}{\ensuremath{\mathbb{R}}}
```

## Пользовательские команды с аргументами

Можно определять и команды с аргументами. Тогда необходимо указать количество аргументов в квадратных скобках, а ссылаться на них внутри команды можно с помощью символа “#” и порядкового номера после неё.

Например, следующее определение

```
% preamble  
\newcommand{\ton}[1]{1, 2, \dots, #1}
```

сокращает запись формул вида

Здесь  $a_{ij}$ ,  $i = \ton{n}$ ,  $j = \ton{m}$  — элементы матрицы  $A$ .

Здесь  $a_{ij}$ ,  $i = 1, 2, \dots, n$ ,  $j = 1, 2, \dots, m$  — элементы матрицы  $A$ .

## Пользовательские команды с аргументами

Запись уравнений в частных производных значительно упрощается, если определить команду `\dd` следующим образом.

```
% preamble  
\newcommand{\dd}[2]{\frac{\partial #1}{\partial #2}}
```

Тогда уравнение  $\frac{\partial \psi}{\partial t} + u \frac{\partial \psi}{\partial x} = 0$  набирается с помощью

```
$\dd{\psi}{t} + u\dd{\psi}{x}=0$
```

вместо

```
$\frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} = 0$
```

## Переопределение команды

Если команда с таким именем уже существует, то попытка переопределить её через `\newcommand` приведет к ошибке. Если все же есть такая необходимость, то можно воспользоваться альтернативной командой `\renewcommand`.

Например, в русскоязычных текстах часто переопределяют команды `\le` и `\ge` на `\leqslant` и `\geqslant` соответственно.

Пусть  $x \geq 0$  и  $y \leq 0$ .

Пусть  $x \geq 0$  и  $y \leq 0$ .

```
% preamble  
\renewcommand{\le}{\leqslant}  
\renewcommand{\ge}{\geqslant}
```

Пусть  $x \geq 0$  и  $y \leq 0$ .

Пусть  $x \geq 0$  и  $y \leq 0$ .

## Таблицы

---

## Таблицы

Для создания таблиц используется окружение `tabular`. Обязательный аргумент окружения определяет количество столбцов, выравнивание текста внутри них и не только. Например, строка “`ccc`” определяет три столбца с выравниванием по центру (“`c`” — “center”).

- Таблица задаётся по строкам;
- Строки отделяются друг от друга комбинацией “`\\"`”;
- Ячейки внутри отделяются символом амперсанда “`&`”

```
\begin{tabular}{ccc}
$T_{11}$ & $T_{12}$ & $T_{13}$\\
$T_{21}$ & $T_{22}$ & $T_{23}$
\end{tabular}
```

$$\begin{array}{ccc} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \end{array}$$

## Таблицы

По умолчанию таблица рисуется без рамок. Символом “|” можно указать вертикальные рамки в преамбуле.

```
\begin{tabular}{|c|c|c|}  
$T_{11}$ & $T_{12}$ & $T_{13}$\\  
$T_{21}$ & $T_{22}$ & $T_{23}$\\  
\end{tabular}
```

$T_{11}$	$T_{12}$	$T_{13}$
$T_{21}$	$T_{22}$	$T_{23}$

Горизонтальные линии рисуются командой `\hline`.

```
\begin{tabular}{|c|c|c|}  
\hline  
$T_{11}$ & $T_{12}$ & $T_{13}$\\  
\hline  
$T_{21}$ & $T_{22}$ & $T_{23}$\\  
\hline  
\end{tabular}
```

$T_{11}$	$T_{12}$	$T_{13}$
$T_{21}$	$T_{22}$	$T_{23}$

# Таблицы

Вместо выравнивания по центру можно указать выравнивание по левому краю символом “l” или по правому краю символом “r”. Количество символов “l”, “c” и “r” должно совпадать с количеством столбцов.

```
\begin{tabular}{|l|c|r|}\hline L & C & R\\\hline LL & CC & RR\\\hline LLL & CCC & RRR\\\hline\end{tabular}
```

L	C	R
LL	CC	RR
LLL	CCC	RRR

# Таблицы

Команда `\multicolumn` позволяет объединять соседние ячейки одной строки между собой. У нее три обязательны аргумента: 1) количество занимаемых столбцов; 2) выравнивание текста для этой ячейки; 3) текст ячейки.

Например, `\multicolumn{3}{|c|}{abc}` — объединит три ячейки таблицы и напишет внутри выровненный по центру текст “abc” с рамками по обеим сторонам.

```
\begin{tabular}{|c|c|c|c|c|}\hline
\multicolumn{2}{|c|}{A} & \multicolumn{2}{|c|}{B} \\
\hline
$x_A$ & $y_A$ & $x_B$ & $y_B$ \\
\hline
\end{tabular}
```

A		B	
$x_A$	$y_A$	$x_B$	$y_B$

# Таблицы

Чтобы объединить строки столбца необходимо подключить пакет `\multirow` и использовать одноименную команду, которая имеет такие же аргументы, но второй из них отвечает за ширину ячейки.

```
\begin{tabular}{|c|c|}\hline
\multirow{3}{4em}{Ячейка в\\ несколько строк} & 1\\
& 2 \\
& 3 \\
\hline
\end{tabular}
```

Ячейка в несколько строк	1
	2
	3

# Таблицы

```
\begin{tabular}{ |c|c|c| }
\hline
Модель & Выборка & Точность \\ \hline
\multirow{2}{5em}{Линейная модель} & train & 0.95 \\ \cline{2-3}
& test & 0.94 \\ \hline
\multirow{2}{5em}{Нейронная сеть} & train & 0.99 \\ \cline{2-3}
& test & 0.5 \\
\hline
\end{tabular}
```

Модель	Выборка	Точность
Линейная модель	train	0.95
	test	0.94
Нейронная сеть	train	0.99
	test	0.5

## Плавающие таблицы

```
\begin{table}
\begin{tabular}{|c|c|}
\hline $T_1$ & $T_2$ \\ \hline
\end{tabular}
\caption{Моя таблица}\label{tab:example}
\end{table}
```

В таблице `\ref{tab:example}` приведены значения величины  $T_i$ ,  $i=1,2$ .

$T_1$	$T_2$
-------	-------

**Таблица 14:** Моя таблица

В таблице 14 приведены значения величины  $T_i$ ,  $i = 1, 2$ .

## Таблицы: пакет csvsimple

```
\csvautotabular{tables/table.csv}
```

Year	Make	Model	Length
1997	Ford	E350	2.35
2000	Mercury	Cougar	2.38

## Графические иллюстрации

---

## Растровые изображения

Чтобы вывести растровое изображение из файла необходимо подключить пакет `graphicx` (`\usepackage{graphicx}`) и команду `\includegraphics`, которая своим обязательным аргументом принимает путь к файлу с изображением. Загруженное изображение воспринимается  $\text{\LaTeX}$  как один большой символ. Необязательный аргумент отвечает позволяет, например, задать размер изображения.

Логотип `\textbf{python}` `\includegraphics[width=1em]{pictures/python.png}` изображает в стиле Майа двух гнездящихся змей.

Логотип `python`  изображает в стиле Майа двух гнездящихся змей.

## Растровые изображения

Необязательный аргумент позволяет задавать размер изображения. `scale` масштабируют относительно исходного изображения. `width` и `height` подгоняют размер изображение под заданную ширину или высоту.

```
\includegraphics[scale=0.5]{pictures/python.png}
\includegraphics[scale=2.0]{pictures/python.png}
\includegraphics[width=1em]{pictures/python.png}
\includegraphics[width=2cm, height=1cm]{pictures/python.png}
\includegraphics[width=2cm, height=1cm,
keepaspectratio]{pictures/python.png}
```



## Растровые изображения

Для того, чтобы растянуть изображение по ширине страницы удобно использовать комбинацию `width=\textwidth`.

```
\includegraphics[width=\textwidth]{pictures/galaxy.jpg}
```



## Плавающие иллюстрации

```
\begin{figure}
    \includegraphics[scale=0.3]{pictures/streamplot.png}
    \caption{Линии тока}\label{fig:streamplot}
\end{figure}
```

На рис. `\ref{fig:streamplot}` построены линии тока функции  $f(x, y)$ .

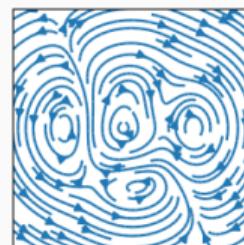
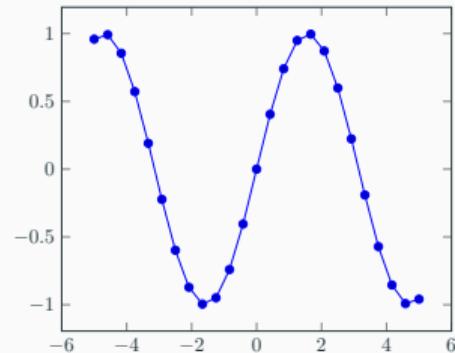


Рис. 1: Линии тока

На рис. 1 построены линии тока функции  $f(x, y)$ .

# Векторная графика: tikzpicture

```
\begin{tikzpicture}
  \begin{axis}
    \addplot {sin(deg(x))};
  \end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
  \begin{axis}
    \addplot3[surf,domain=0:360,samples=40]
    {sin(x)*sin(y)};
  \end{axis}
\end{tikzpicture}
```

