

# Программирование графики (окончание)

Учебная практика

мехмат, III курс, кафедра ИВЭ

Стили

Параметры

Стиль

Пример 1

Пример 2

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Определение параметров команд и стили

# Места определения параметров команд

Стили

**Параметры**

Стиль

Пример 1

Пример 2

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Параметры могут указываться:

- ✓ Непосредственно в командах.

Пример

```
\draw [->] (0, 0) -- (1, 0);
```

- ✓ В областях, в частности, в окружении `tikzpicture`.

Пример

```
{ [->] %  
  \draw (0, 0) -- (1, 0);  
  \draw (0, 1) -- (1, 1);  
} %
```

- ✓ В *стилях*.

# Стиль

- Стили
- Параметры
- Стиль**
- Пример 1
- Пример 2
- Стандартные Пример
- Матрицы 1
- Матрицы 2
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2

**Стиль:** именованный набор настроек.

`<имя_стиля>/.style = <параметры>`

Определяет или переопределяет стиль с заданным именем.

`<имя_стиля>/.append style = <параметры>`

Дополняет стиль с заданным именем.

`<имя_стиля>`

Исполняет настройки, записанные в заданном стиле.

*Замечание:* при помощи переопределений и дополнений стилей можно включать/отключать цвета, эффекты теней и т. д. в разных версиях документа, создаваемого на основе пакета beamer (презентация, статья, раздаточный материал и т. д.) △

# Пример определения стиля

Стили

Параметры

Стиль

**Пример 1**

Пример 2

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

```
\usepackage {tikz}

\usetikzlibrary {shadows}

\tikzset %
{ %
  box/.style = %
  { %
    shape = rectangle, %
    draw, %
    % ...
  } %
}
```

```
\mode <beamer> %
{ %
  \tikzset %
  { %
    box/.append style = %
    { %
      draw = structure, %
      fill = %
      structure!30!white, %
      drop shadow %
    } %
  } %
}
```

# Пример использования стиля

Стили

Параметры

Стиль

Пример 1

**Пример 2**

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

```
\begin {frame} {Рисунок}
  \begin {figure}
    \begin {tikzpicture}
      \node [box] (A) {Объект};
      % ...
    \end {tikzpicture}
    \caption {диаграмма}
  \end {figure}
\end {frame}
```

# Предопределённые стили

- Стили
- Параметры
- Стиль
- Пример 1
- Пример 2
- Стандартные**
- Пример
- Матрицы 1
- Матрицы 2
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2

Таблица 1: основные предопределённые стили

Стиль	Исполняется в начале каждой команды/окружения/ключа
<b>every picture</b>	tikz, tikzpicture
<b>every scope</b>	scope
<b>every path</b>	path
<b>every node</b>	node
<b>every edge</b>	edge
<b>every matrix</b>	matrix
<b>every cell</b>	Ячейка матрицы
<b>every on chain</b>	on chain
<b>every join</b>	join
	...

# Пример использования predefined styles

Стили

Параметры

Стиль

Пример 1

Пример 2

Стандартные

**Пример**

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

```
\tikzset %  
{ %  
  every picture/.style = %  
  { %  
    thick, %  
  } %  
} %
```

```
\begin {tikzpicture} %  
[ %  
  every node/.style = %  
  { %  
    draw, %  
    font = \large %  
  } %  
] %  
\node %  
[ %  
  fill = gray, %  
  font = \small %  
] %  
{Text};  
\end {tikzpicture}
```



# Стиль вершин матрицы

Стили

Параметры

Стиль

Пример 1

Пример 2

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**nodes** =  $\langle \text{параметры} \rangle$

Дополняет стиль всех вершин, находящихся в текущей матрице. Сокращённая запись для `every node/.append style =  $\langle \text{параметры} \rangle$` .

**cells** =  $\langle \text{параметры} \rangle$

Аналогично, для ячеек:

`every cell/.append style =  $\langle \text{параметры} \rangle$` .

**row**  $\langle \text{номер} \rangle$  =  $\langle \text{параметры} \rangle$

**column**  $\langle \text{номер} \rangle$  =  $\langle \text{параметры} \rangle$

**row**  $\langle \text{строка} \rangle$  **column**  $\langle \text{столбец} \rangle$  =  $\langle \text{параметры} \rangle$

Аналогично, для строки/столбца/ячейки с заданным номером.

# Стили ячеек матрицы

Стили

Параметры

Стиль

Пример 1

Пример 2

Стандартные

Пример

Матрицы 1

Матрицы 2

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

## **every odd/even row/column**

Устанавливают стиль для ячеек нечётной/чётной строки/столбца.

### Пример

```
\matrix [matrix of math nodes, %  
  every even row/.append style =  
    {nodes = {fill = orange}}, %  
  every even column/.append style =  
    {text = green!50!black}] %  
{ %  
  a_{1, 1} & a_{1, 2} & %  
  a_{1, 3} & a_{1, 4} \\  
  % ...  
};%
```

$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$
$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$
$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$
$a_{4,1}$	$a_{4,2}$	$a_{4,3}$	$a_{4,4}$

Стили

**Деревья**

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

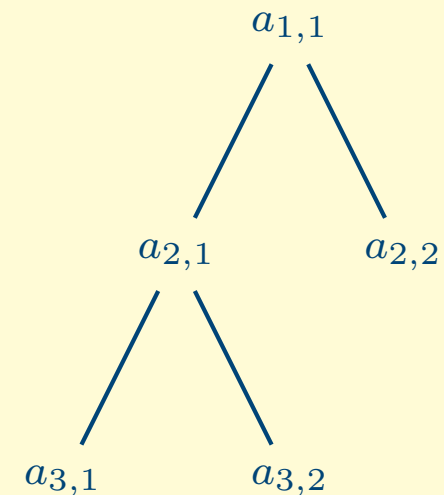
# Отображение древовидных структур

# Определение деревьев в коде

- ✓ При помощи синтаксиса вершины можно определить подчинённую ей иерархию вершин.
- ✓ Пакет `tikz-qtrees` предоставляет более компактный по сравнению с `TikZ` синтаксис для определения деревьев.

## Пример

```
\node {$a_{1, 1}$} %  
  child %  
  { %  
    node {$a_{2, 1}$} %  
      child { node {$a_{3, 1}$} } %  
      child { node {$a_{3, 2}$} } %  
    } %  
  child { node {$a_{2, 2}$} };%
```



Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Синтаксис операции пути `child`

**child** [*⟨параметры⟩*] **foreach** *⟨переменные⟩* **in** {*⟨значения⟩*}  
*{⟨дочерний\_путь⟩*}

Операция должна следовать вдоль пути непосредственно за законченной операцией **node** или **child**. Выполняет отображение всех иерархически подчинённых путей, соединяя их дугами. При выводе каждого дочернего пути начало координат устанавливается в вычисленную позицию этого пути. Если дочерний путь пропущен, или в нём отсутствует вершина, в соответствующую позицию автоматически вставляется пустая вершина формы **coordinate**.

Стили

Деревья

Вершины

**Синтаксис**

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Пропуск дочерних вершин

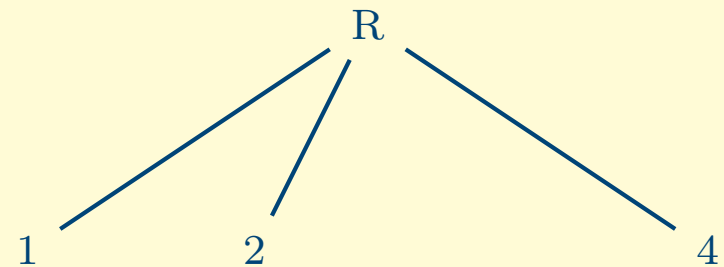
**missing** = true, false

(по умолчанию: **false**)

Указывается в параметрах к `child`. Означает, что соответствующая дочерняя вершина будет учтена при вычислении позиции остальных, но операции отображения её самой исполнены не будут.

## Пример

```
\node {R} %  
  child { node {1} } %  
  child { node {2} } %  
  child [missing] %  
    { node {3} } %  
  child { node {4} };
```



Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Расположение вершин дерева

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**level distance** =  $\langle \text{длина} \rangle$  (по умолчанию: 15mm)

Определяет расстояние от родителя до прямой, на которой располагаются дочерние узлы.

**sibling distance** =  $\langle \text{длина} \rangle$  (по умолчанию: 15mm)

Определяет расстояние между якорями дочерних узлов.

**growth parent anchor** =  $\langle \text{имя\_якоря} \rangle$  (по умолчанию: **center**)

Определяет якорь родительской вершины, относительно которого будет вычислены позиции дочерних.

**grow** =  $\langle \text{направление} \rangle$  (по умолчанию: **down**)

Определяет направление «роста» дерева. Может быть целым числом (градусы) или таким значением, как **up**, **south west** и т. д.

# Расположение вершин дерева (окончание)

- Стили
- Деревья
  - Вершины
  - Синтаксис
  - Пропуск
  - Расположение 1
  - Расположение 2**
  - Пример 1
  - Замечания
  - Имена
  - Дуги 1
  - Пример 2
  - Дуги 2
  - Дуги 3
  - Пример 3
  - Формы дуг
  - Стили 1
  - Стили 2
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2

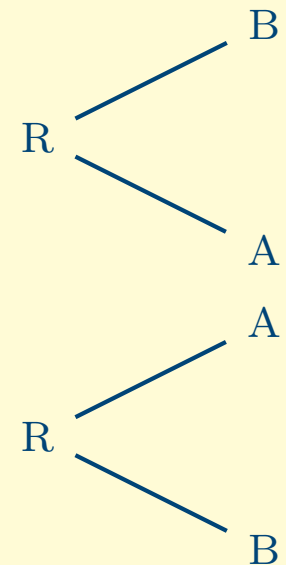
**grow'** = *⟨направление⟩* (по умолчанию: **down**)

Аналогично, только дочерние вершины располагаются в обратном порядке.

## Пример

```
\node (A) {R} %  
  [grow = right] %  
  child { node {A} } %  
  child { node {B} };
```

```
\node [below = 15mm of A] {R} %  
  [grow' = right] %  
  child { node {A} } %  
  child { node {B} };
```

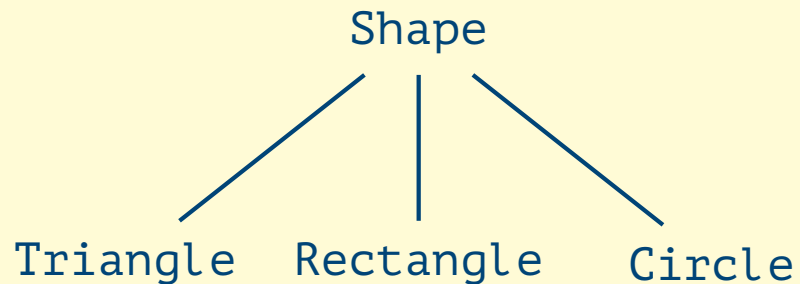




# Пример определения дочерних узлов при помощи операции `foreach`

## Пример

```
\node [font = \ttfamily] {Shape} %  
  [font = \ttfamily, %  
  level distance = 4em, %  
  sibling distance = width("Rectangle") + 1em] %  
  child foreach \name in {Triangle, Rectangle, Circle} %  
  { %  
    node {\name} %  
  };%
```



Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

**Пример 1**

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Замечания к примеру

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

**Замечания**

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

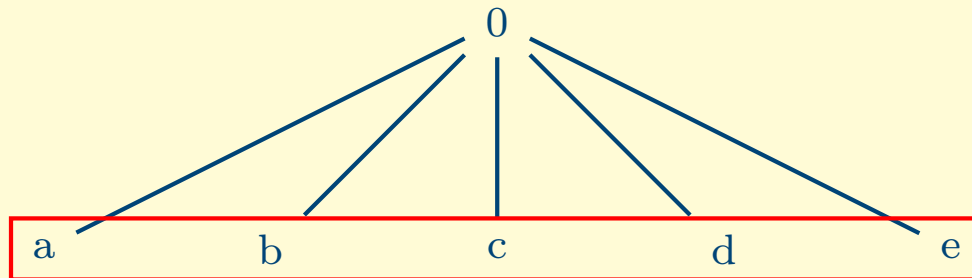
- ✓ Функции `width("str")`, `height("str")` и `depth("str")` возвращают, соответственно, ширину, высоту и глубину (высота от базовой линии текста до его нижнего края) строки `str`, если бы она была набрана текущим шрифтом. Функции являются частью математического модуля `TikZ` (см. далее).
- ✓ Настройки перед операцией `child` относятся к дочерней вершине и всем подчинённым ей.
- ✓ Вместо функции роста деревьев по умолчанию (использующей настройки `level distance` и т. д.) можно определять собственные функции. Некоторые дополнительные готовые функции роста определены в библиотеке `TikZ trees`.

# Имена дочерних вершин

*Замечание:* для всех дочерних вершин, имена которым не были определены пользователем, автоматически назначаются имена вида:  $\langle \text{имя\_родителя} \rangle - \langle \text{номер} \rangle$  (рекурсивно для нижних уровней:  $\langle \text{имя\_родителя} \rangle - \langle \text{номер}_1 \rangle - \langle \text{номер}_2 \rangle$  и т. д.) △

## Пример

```
\node (T) {0} %  
  child foreach \i in {a, ..., e} {node {\i}};  
\draw [red] (T-1.north west) rectangle (T-5.south east);
```



- Стили
- Деревья
- Вершины
- Синтаксис
- Пропуск
- Расположение 1
- Расположение 2
- Пример 1
- Замечания
- Имена**
- Дуги 1
- Пример 2
- Дуги 2
- Дуги 3
- Пример 3
- Формы дуг
- Стили 1
- Стили 2
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2

# Операция рисования дуги от родительской к дочерней вершине

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**edge from parent** [*⟨параметры⟩*]

Определяет параметры отображения дуги. Эта операция должна встречаться только внутри дочернего пути в конце, возможно, после описания дочерней вершины. Возможно указание после неё операций **node**, которые будут добавлены вдоль дуги. Если эта операция пропущена, будет автоматически добавлена дуга, использующая следующий стиль:

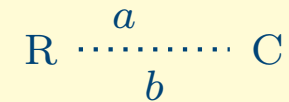
**edge from parent** (стиль, по умолчанию: draw)

Исполняется перед рисованием дуги по умолчанию.

# Пример расположения вершин вдоль дуги от родительской к дочерней вершине

## Пример

```
\node {R} %  
  [grow = right] %  
  child %  
  { %  
    node {C} %  
    edge from parent [dotted] %  
      node [pos = .3, above] {$a$} %  
      node [below] {$b$} %  
  };
```



Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

**Пример 2**

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

# Операция рисования дуги от родительской к дочерней вершине (продолжение)

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**edge from parent path** =  $\langle \text{путь} \rangle$  (по умолчанию: см. ниже)

Определяет операции рисования дуги. По умолчанию содержит код:

```
(\tikzparentnode \tikzparentanchor) -- %  
(\tikzchildnode \tikzchildanchor)
```

Здесь вместо макросов `\tikzparentnode` и т. д. при рисовании будут подставлены имя родительской вершины, её якорь (при необходимости с точкой в начале) и имя с якорем дочерней. Таким образом, по умолчанию будет нарисована прямая линия.

*Замечание:* далее будут приведены готовые операции рисования дуг из библиотеки TikZ trees.



# Операция рисования дуги от родительской к дочерней вершине (окончание)

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**parent anchor** =  $\langle \text{имя\_якоря} \rangle$  (по умолчанию: **border**)

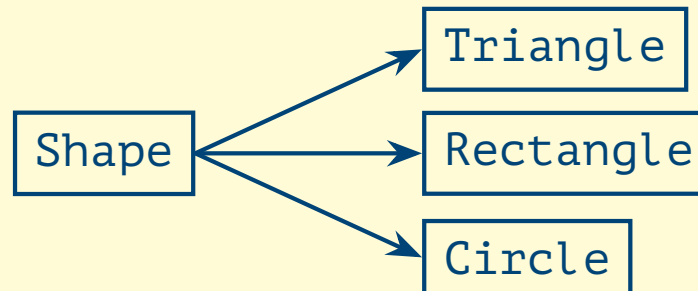
**child anchor** =  $\langle \text{имя\_якоря} \rangle$  (по умолчанию: **border**)

Определяют якоря родительской и дочерней вершины, между которыми рисуется дуга. Значение **border** приводит к передаче пустой строки вместо макроса `\tikzparentanchor` или `\tikzchildanchor`, в результате чего дуга будет начинаться/заканчиваться на границе вершины.

# Пример определения вида дуг между родительскими и дочерними вершинами

## Пример

```
\tikzset { %  
  every node/.style = {draw, font = \ttfamily}, %  
  edge from parent/.append style = {-Stealth} } %  
\node {Shape} %  
  [sibling distance = 1.8em, %  
  grow' = right, growth parent anchor = east, %  
  parent anchor = east, child anchor = west] %  
  child foreach \name in {Triangle, Rectangle, Circle} %  
    { node [anchor = west] {\name} };
```



Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

**Пример 3**

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2



# Предопределённые пути для рисования дуг библиотеки TikZ trees

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

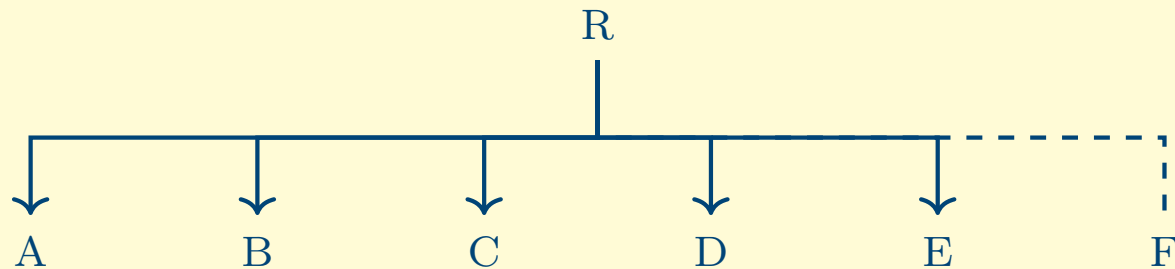
Графики 2

**edge from parent fork down/right/left/up**

(стиль)

Пример

```
\node {R} %  
[->, edge from parent fork down]  
child foreach \i in {A, ..., E} { node {\i} } %  
child { node {F} edge from parent [-, dashed] };
```



# Стили, относящиеся к деревьям

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**every child** (стиль, по умолчанию: *⟨пусто⟩*)

Исполняется перед отображением пути каждой дочерней вершины, аналогично указываемым параметрам операции `child`.

**every child node** (стиль, по умолчанию: *⟨пусто⟩*)

Исполняется перед отображением каждой дочерней вершины, в дополнение к стилю `every node`.

**level** = *⟨номер⟩* (стиль, по умолчанию: *⟨пусто⟩*)

Исполняется перед отображением дочерних вершин одной родительской, *⟨номер⟩* определяет слой дерева. При переопределении стиля этот номер доступен через параметр «`#1`».

# Стили, относящиеся к деревьям

Стили

Деревья

Вершины

Синтаксис

Пропуск

Расположение 1

Расположение 2

Пример 1

Замечания

Имена

Дуги 1

Пример 2

Дуги 2

Дуги 3

Пример 3

Формы дуг

Стили 1

Стили 2

Вычисления 1

Вычисления 2

Графики 1

Графики 2

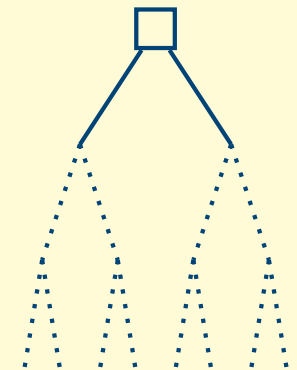
**level** *⟨номер⟩*

(стиль, по умолчанию: *⟨пусто⟩*)

Исполняется перед отображением дочерних вершин заданного уровня, в дополнение к стилю level.

## Пример

```
\tikzset { %  
  level/.style = {sibling distance = 20mm / 2^#1},  
  level 2/.style = {dotted} } %  
\node [draw] {} %  
  [level distance = 2em] %  
  child foreach \i in {1, 2} %  
    { child foreach \j in {1, 2} %  
      { child foreach \k in {1, 2} } %  
    } %  
};
```



Стили

Деревья

**Вычисления 1**

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

# Вычисления координат

# Обзор средств определения координат

Стили

Деревья

Вычисления 1

**Обзор**

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

Для определения сложных правил расположения графических объектов пакет `TikZ` предоставляет следующие возможности:

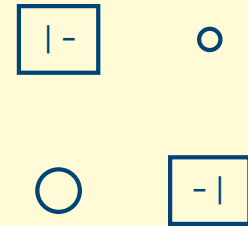
- ✓ операции «|-» и «-|» для нахождения точек пересечения горизонтальной и вертикальной прямых, проходящих через точки с заданными координатами.
- ✓ встроенные средства преобразования координат при помощи параметров (сдвиг, поворот, и т. д.);
- ✓ библиотеку `TikZ calc` для вычисления координат;
- ✓ операцию пути `let`;
- ✓ математический модуль `TikZ`.

Все перечисленные модули можно использовать как по отдельности, так и совместно друг с другом.

# Нахождение координат пересечения горизонтальной и вертикальной прямой

## Пример

```
\draw (0, 0) circle [radius = 4pt] %  
  coordinate (A);  
\draw (1, 1) circle [radius = 2pt];  
\node [draw] at (A |- 1, 1) {\texttt {|-}};  
\node [draw] at (A -| 1, 1) {\texttt {-|}};
```



*Замечание:* операции над координатами «|-» и «-|» не следует путать с рассмотренными ранее одноимёнными операциями путей, которые добавляли к пути пары отрезков. △

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

## Математический модуль (слой, уровень) TikZ:

предоставляет возможности вычисления по формулам, записанным на интуитивно понятном языке, в вещественной арифметике над скалярами и длинами. Так как реализован на математических возможностях  $\text{T}_\text{E}_\text{X}$ , значения должны находиться в диапазоне  $\pm 16\,383,999\,99$ . Может использоваться как неявно в высокоуровневых конструкциях TikZ, когда в аргументах команд вместо значений указываются выражения, так и отдельно от PGF при помощи команд `\pgfmathparse` и т. д.

*Замечание:* ранее рассмотренный пример изображения дерева использовал математический модуль для определения расстояния между вершинами в зависимости от уровня (с операциями деления и возведения в степень).



# Синтаксис выражений, поддерживаемых математическим модулем TikZ

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

**Выражения**

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

Математический модуль поддерживает следующие элементы выражений в инфиксной записи:

- ✓ десятичные вещественные числа в экспоненциальной записи;
- ✓ целые числа в двоичной, восьмеричной и шестнадцатеричной записи;
- ✓ регистры и размерности  $\text{T}_{\text{E}}\text{X}$ ;
- ✓ текстовые строки в двойных кавычках;
- ✓ арифметические операции и функции;
- ✓ скобки.



# Операции и функции, поддерживаемые математическим модулем

Таблица 2: основные операции математического модуля

$-x$	$x + y$	$x - y$	$x * y$	$x / y$	$x ^ y$
$x == y$	$x != y$	$x > y$	$x < y$	$x >= y$	$x <= y$
$!x$	$x \&\& y$	$x    y$	$x ? y : z$		
$xr$ (преобразование радианов к градусам)					

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

**Операции 1**

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

# Операции и функции, поддерживаемые математическим модулем (окончание)

Таблица 3: основные функции математического модуля

$\text{sqrt}(x)$	$\text{pow}(x, y)$	$e$	$\text{exp}(x)$
$\ln(x)$	$\log_2(x)$	$\log_{10}(x, y)$	$\text{abs}(x)$
$\text{mod}(x, y)$	$\text{round}(x)$	$\text{floor}(x)$	$\text{ceil}(x)$
$\sin(x)$	$\cos(x)$	$\tan(x)$	$\pi$
$\text{asin}(x)$	$\text{acos}(x)$	$\text{atan}(x)$	$\text{atan2}(x, y)$
$\text{rnd} (\in [0, 1])$		$\text{random}(x, y) (\in [x, y])$	
$\text{vecLen}(x, y)$ (длина двумерного вектора)			
$\text{width}("s")$	$\text{height}("s")$	$\text{depth}("s")$	

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

**Операции 2**

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

# Встроенные средства TikZ для преобразования координат

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

**Преобразования**

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

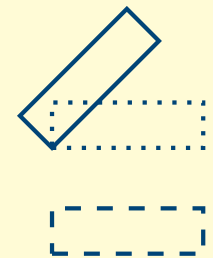
Графики 1

Графики 2

**Средства преобразования координат:** аналогично библиотеке OpenGL, позволяют определять последовательность преобразований для используемых координат (и только для них, но не для текста, толщины линий и т. п.) На координаты, расположенные внутри последовательно вложенных областей с различными матрицами преобразований, действует общая матрица, равная произведению всех матриц областей.

## Пример

```
\draw [dashed] (0, 0) rectangle (1, 0.3);  
{ [yshift = 0.7cm] %  
  \draw [dotted] (0, 0) rectangle (1, 0.3);  
  \draw [rotate = 45] (0, 0) rectangle (1, 0.3);  
} %
```



# Основные параметры преобразований координат

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

**Настройки 1**

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

**xshift** =  $\langle \text{длина} \rangle$

**yshift** =  $\langle \text{длина} \rangle$

Сдвигает координаты на заданную длину вдоль, соответственно, оси  $x$  и  $y$ .

**shift** =  $\{ \langle \text{координата} \rangle \}$

Сдвигает координаты на величины, равные  $x$ - и  $y$ -компонентам заданной координаты, одновременно вдоль осей  $x$  и  $y$ .

# Основные параметры преобразований координат (продолжение)

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

**scale** =  $\langle \text{множитель} \rangle$

Умножает все координаты на заданный множитель.

**xscale** =  $\langle \text{множитель} \rangle$

**yscale** =  $\langle \text{множитель} \rangle$

Умножает только, соответственно,  $x$ - и  $y$ -компоненты координат на заданный множитель

**scale around** =  $\{ \langle \text{множитель} \rangle : \langle \text{координата} \rangle \}$

Выполняет масштабирование координат. В отличие от `scale` центром масштабирования является не начало координат, а заданная координата.

# Основные параметры преобразований координат (окончание)

Стили

Деревья

Вычисления 1

Обзор

Пересечение

Математика

Выражения

Операции 1

Операции 2

Преобразования

Настройки 1

Настройки 2

Настройки 3

Вычисления 2

Графики 1

Графики 2

**rotate** =  $\langle \text{угол} \rangle$

Выполняет поворот координат против часовой стрелки на заданный угол в градусах относительно начала координат.

**rotate around** =  $\{ \langle \text{угол} \rangle : \langle \text{координата} \rangle \}$

Поворачивает координаты относительно заданной точки.

**rotate around x** =  $\langle \text{угол} \rangle$

**rotate around y** =  $\langle \text{угол} \rangle$

**rotate around z** =  $\langle \text{угол} \rangle$

Поворачивают трёхмерные координаты на заданный угол вокруг оси  $x$ ,  $y$  и  $z$  соответственно.

Стили

Деревья

Вычисления 1

**Вычисления 2**

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

Пример

let 1

let 2

let 2

Графики 1

Графики 2

# Вычисления координат (окончание)

# Выражения библиотеки calc

Стили

Деревья

Вычисления 1

Вычисления 2

**calc**

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

Пример

let 1

let 2

let 2

Графики 1

Графики 2

После подключения библиотеки TikZ calc становятся доступны выражения, заключённые в символы « $\langle \$ \dots \$ \rangle$ », при помощи которых можно выполнять следующие операции с координатами:

- ✓ сумма и разность координат (то есть смещение первой координаты на величину второй);
- ✓ умножение координаты на число (масштабирование относительно начала координат);
- ✓ Нахождение координат точки на прямой, проходящей через две другие точки с заданными координатами, которая либо находится на заданном расстоянии от одной из них, либо делит отрезок в заданной пропорции, либо является проекцией на прямую третьей точки.



# Синтаксис выражений пакета calc

Стили

Деревья

Вычисления 1

Вычисления 2

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

Пример

let 1

let 2

let 2

Графики 1

Графики 2

- ✓ Синтаксис выражения координат:

$([\langle \text{параметры} \rangle] \$ \langle \text{координатное\_выражение} \rangle \$)$

- ✓  $\langle \text{параметры} \rangle$  могут содержать ранее рассмотренные настройки преобразования координат, такие как xshift.
- ✓  $\langle \text{координатное\_выражение} \rangle$  имеет следующий синтаксис:

$\langle \text{компонент} \rangle \{ \langle \text{операция} \rangle \langle \text{компонент} \rangle \}$

- ✓  $\langle \text{операция} \rangle$  может быть «+» или «-».
- ✓  $\langle \text{компонент} \rangle$  имеет следующий синтаксис:

$\langle \text{множитель} \rangle * \langle \text{координата}_1 \rangle \langle \text{модификаторы} \rangle$

# Синтаксис выражений пакета calc (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- calc
- Синтаксис 1
- Синтаксис 2**
- Синтаксис 3
- Синтаксис 4
- Синтаксис 5
- Пример
- let 1
- let 2
- let 2
- Графики 1
- Графики 2

- ✓  $\langle \text{множитель} \rangle$  может представлять собой произвольное выражение, поддерживаемое математическим модулем TikZ. Считается, что множитель заканчивается до следующего вхождения символов «\*(» (без пробела). Так как сам множитель может содержать скобки, для разрешения неоднозначностей возможно поместить его внутрь фигурных скобок.
- ✓  $\langle \text{модификаторы} \rangle$  имеют следующий синтаксис:

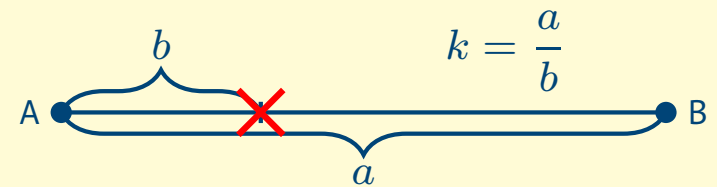
$! \langle \text{параметр} \rangle ! \langle \text{угол} \rangle : \langle \text{координата}_2 \rangle$
- ✓ Необязательный аргумент  $\langle \text{угол} \rangle$  может задавать угол поворота. Он также может задаваться выражением, поддерживаемым математическим модулем TikZ.
- ✓ Все координаты должны заключаться в круглые скобки.

# Синтаксис выражений пакета calc (продолжение)

- ✓ В зависимости от типа модификатора *⟨параметр⟩* может представлять собой выражение множителя, длины или третьей координаты.

## Множитель

$(\$ (A) ! k ! (B) \$)$



## Длина

$(\$ (A) ! d ! (B) \$)$



- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- calc
- Синтаксис 1
- Синтаксис 2
- Синтаксис 3**
- Синтаксис 4
- Синтаксис 5
- Пример
- let 1
- let 2
- let 2
- Графики 1
- Графики 2

# Синтаксис выражений пакета calc (продолжение)

Стили

Деревья

Вычисления 1

Вычисления 2

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

**Синтаксис 4**

Синтаксис 5

Пример

let 1

let 2

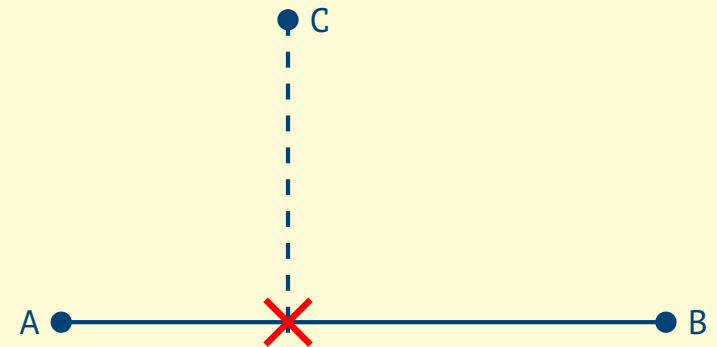
let 2

Графики 1

Графики 2

Координата

$(\$ (A) ! (C) ! (B) \$)$

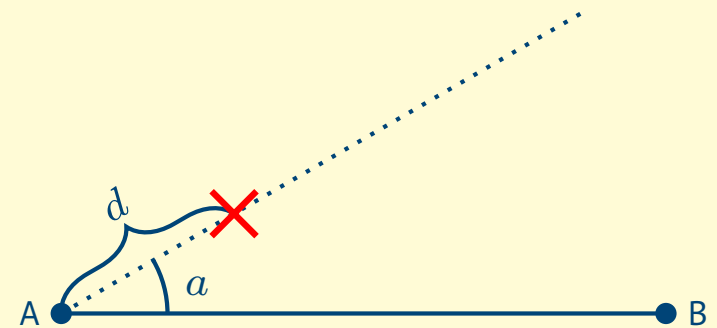


# Синтаксис выражений пакета calc (окончание)

- ✓ Если в модификаторах присутствует угол, на него поворачивается прямая, на которой расположена искомая точка, относительно первой из двух заданных точек.

+ угол

$(\$ (A) ! d ! a : (B) \$)$



- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- calc
- Синтаксис 1
- Синтаксис 2
- Синтаксис 3
- Синтаксис 4
- Синтаксис 5**
- Пример
- let 1
- let 2
- let 2
- Графики 1
- Графики 2

# Пример вычислений координат

Стили

Деревья

Вычисления 1

Вычисления 2

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

**Пример**

let 1

let 2

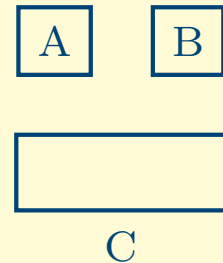
let 2

Графики 1

Графики 2

## Пример

```
\node [draw] (A) {A};
\node [draw, right = 1em of A] (B) {B};
\coordinate (D) at %
  ([rotate = -90]$(B.west) - (A.east)$);
\draw %
  ($(A.south west) + (D)$) %
  coordinate (E) %
  rectangle %
  ($(B.south east) + (D) - (0, .5cm)$) %
  coordinate (F);
\coordinate (G) at (E |- F);
\node [anchor = north] at ($(G)!.5!(F)$) {C};
```



# Операция пути let

- ✓ Синтаксис операции пути let:

```
let <присваивание> { , <присваивание> } in <тело>
```

- ✓ <присваивание> может выполняться с вычислением скаляра (длины) или координаты. В зависимости от этого оно может иметь одну из двух форм:

```
\n<скалярный_регистр> = {<выражение>}
```

```
\p<координатный_регистр> = {<выражение>}
```

- ✓ Имя регистра может быть либо целым неотрицательным числом, либо строкой в фигурных скобках.

Стили

Деревья

Вычисления 1

Вычисления 2

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

Пример

let 1

let 2

let 2

Графики 1

Графики 2

# Операция пути `let` (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- calc
- Синтаксис 1
- Синтаксис 2
- Синтаксис 3
- Синтаксис 4
- Синтаксис 5
- Пример
- let 1
- let 2**
- let 2
- Графики 1
- Графики 2

- ✓  $\{\langle \text{выражение} \rangle\}$  должно поддерживаться математическим модулем `TikZ` или быть выражением координат соответственно.
- ✓  $\langle \text{тело} \rangle$  может быть определением пути `TikZ`.
- ✓ Скалярное присваивание определяет макрос  $\backslash n\{\langle \text{имя\_регистра} \rangle\}$ , которое хранит результат вычисления своего выражения.
- ✓ Координатное присваивание определяет макросы  $\backslash x\{\langle \text{имя\_регистра} \rangle\}$  и  $\backslash y\{\langle \text{имя\_регистра} \rangle\}$ , которые хранят, соответственно,  $x$ - и  $y$ -координаты результата вычисления своего выражения, а также макрос  $\backslash r\{\langle \text{имя\_регистра} \rangle\}$ , который хранит обе координаты через запятую.

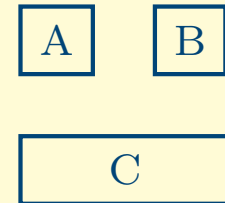


# Операция пути let (окончание)

- ✓ Все вышеперечисленные макросы определены только в последующих присваиваниях и в теле операции let. Обращение к ним заменяется хранящимся в них значением.

## Пример

```
\node [draw] (A) {A};  
\node [draw, right = 1em of A] (B) {B};  
\coordinate (D) at %  
  ([rotate = -90]$(B.west) - (A.east)$);  
\path %  
  let \p1 = ($(B.east) - (A.west)$), %  
      \n1 = {veclen(\x1, \y1)} in %  
  node at ($(A.south west) + (D)$) %  
    [draw, anchor = north west, %  
    minimum width = \n1] {C};
```



Стили

Деревья

Вычисления 1

Вычисления 2

calc

Синтаксис 1

Синтаксис 2

Синтаксис 3

Синтаксис 4

Синтаксис 5

Пример

let 1

let 2

let 2

Графики 1

Графики 2

Стили

Деревья

Вычисления 1

Вычисления 2

**Графики 1**

Обзор

Этапы

Окружения

Пример 1

Команды 1

Команды 2

Выражение 1

Выражение 2

Выражение 3

Координаты 1

Координаты 2

Пример 2

Таблица 1

Таблица 2

Таблица 3

Таблица 4

Графики 2

# Визуализация данных в виде графиков

# Обзор средств рисования графиков

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор**
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

В TikZ доступны следующие средства рисования графиков:

- ✓ ранее рассмотренные низкоуровневые операции с путями;
- ✓ библиотека TikZ datavisualization;
- ✓ внешний пакет  $\text{\LaTeX}$  pgfplots, использующий TikZ (рассматривается далее).

Основные возможности пакета pgfplots:

- ✓ графики: в 2 и 3 измерениях; в декартовых и полярных координатах, обычные, логарифмические и полулогарифмические;
- ✓ внешний вид отображения графиков: линии, точки, кусочно-постоянные прямые, столбиковые диаграммы, сетки, поверхности и т. д.;
- ✓ дополнительные данные: обозначения осей, отметки значений, легенды и т. д.

# Основные этапы работы с пакетом pgfplots

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Обзор

**Этапы**

Окружения

Пример 1

Команды 1

Команды 2

Выражение 1

Выражение 2

Выражение 3

Координаты 1

Координаты 2

Пример 2

Таблица 1

Таблица 2

Таблица 3

Таблица 4

Графики 2

1. Подключение пакета:

```
\usepackage {pgfplots}
```

2. Необязательно: загрузка в преамбуле вспомогательных библиотек pgfplots командой `\usepgfplotslibrary`.

3. Установка в любом месте глобальных настроек при помощи команды `\pgfplotsset`. Как минимум рекомендуется устанавливать параметр совместимости с используемой версией пакета:

```
\pgfplotsset { compat = 1.13, ... }
```

4. Использование команд и окружений пакета внутри команды `TikZ \tikz` или окружения `tikzpicture`.

# Окружения пакета

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения**
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

```
\begin{axis} [параметры]  
  содержимое_окружения  
\end{axis}
```

Отображает оси координат и предназначена для помещения внутрь команд вывода графиков, легенд и т. д.

*Замечание:* вспомогательные окружения **semilogaxis**, **semilogyaxis** и **loglogaxis** для полу- и логарифмических осей являются сокращённой записью использования окружения `axis` со следующими аргументами:

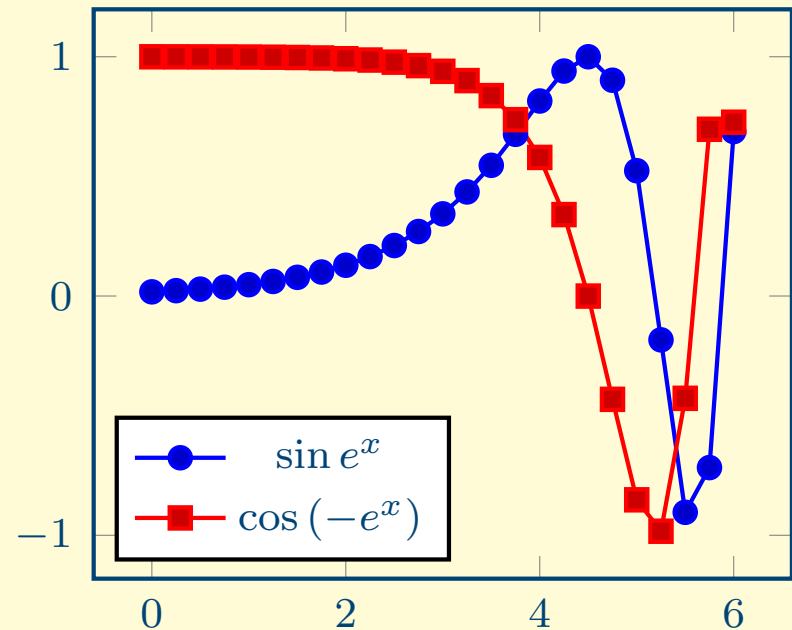
```
\begin {axis} [xmode = normal | log, ymode = normal | log, ...]  
  % ...  
\end {axis}
```



# Пример вывода графиков

## Пример

```
\begin {tikzpicture}
  \begin {axis} %
    [ %
      width = 62mm, %
      domain = 0 : 6, %
      legend pos = south west
    ] %
    \addplot {sin(exp(x))};
    \addplot {cos(-exp(x))};
    \legend %
      { $\sin e^x$ }, %
       $\cos(-e^x)$ 
    \end {axis}
\end {tikzpicture}
```



- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1**
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

# Команды пакета pgfplots

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1**
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

**\addplot** [*⟨параметры⟩*] *⟨входные\_данные⟩*  
*⟨завершающие\_команды\_пути⟩*

Выводит график. Различные варианты указания входных данных будут рассмотрены далее. *⟨параметры⟩* задают внешний вид. Если они пропущены, они загружаются из следующего элемента *циклического списка*, благодаря которому разные графики на одних осях выглядят по-разному. Текущие настройки графика сохраняются для дальнейшего отображения в легенде. *⟨завершающие\_команды\_пути⟩* могут содержать команды пути TikZ, исполняемые в конце вывода графика.

# Команды пакета pgfplots

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2**
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

## Замечания:

- ✓ Команда `\addplot+` отличается от `\addplot` только тем, что её параметры всегда добавляются к параметрам из циклического списка.
- ✓ Аналогично, в пакете определены команды `\addplot3` и `\addplot3+` для построения трёхмерных графиков. △

## `\legend {⟨список⟩}`

Определяет список подписей к элементам легенды, разделённый запятыми. Также включает её отображение.

*Замечание:* рекомендуется не разделять элементы списка легенды запятыми с пробелами, так как пробелы могут попасть в их надписи. △



# Определения источника данных из выражения

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
  - Обзор
  - Этапы
  - Окружения
  - Пример 1
  - Команды 1
  - Команды 2
  - Выражение 1**
  - Выражение 2
  - Выражение 3
  - Координаты 1
  - Координаты 2
  - Пример 2
  - Таблица 1
  - Таблица 2
  - Таблица 3
  - Таблица 4
- Графики 2

**expression** {⟨выражение⟩} (входные данные)

Определяет входные данные в форме выражения, которое затем используется для вычисления значений функции математическим модулем TikZ. ⟨выражение⟩ может зависеть от переменных «x» и «y» (для трёхмерных графиков).

{⟨выражение⟩} (входные данные)

Сокращённая запись для «expression {⟨выражение⟩}».

(⟨выражение\_x⟩, ⟨выражение\_y⟩) (входные данные)

(⟨выражение\_x⟩, ⟨выражение\_y⟩, ⟨выражение\_z⟩)

Аналогично, определяют координаты параметрическим способом.

# Определения источника данных из выражения (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2**
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

**domain** =  $\langle x_{\min} \rangle : \langle x_{\max} \rangle$  (по умолчанию: -5 : 5)

**y domain** =  $\langle y_{\min} \rangle : \langle y_{\max} \rangle$  (по умолчанию: -5 : 5)

Ограничивают диапазоны выводимых значений по осям  $x$  и  $y$ .

**samples** =  $\{\langle \text{количество} \rangle\}$  (по умолчанию: 25)

**samples y** =  $\{\langle \text{количество} \rangle\}$  (по умолчанию: 25)

Определяет количество значений переменной  $x$  ( $y$  для трёхмерного случая), в которых будут вычислены значения координат.

**samples at** =  $\{\langle \text{список\_значений} \rangle\}$

Определяет значения переменной в явном виде.

# Определения источника данных из выражения (окончание)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3**
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

**variable** = {⟨имя\_переменной⟩} (по умолчанию: x)

**variable y** = {⟨имя\_переменной⟩} (по умолчанию: y)

Определяют используемые в выражениях имена переменных вместо «x» и «y».

*Замечание:* по умолчанию библиотека `pgfplots` использует собственный модуль, реализующий вычисления над вещественными числами в двойной точности, вместо рассмотренного ранее математического модуля `TikZ`, точность которого гораздо ниже. △

# Определения источника данных из списка

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
  - Обзор
  - Этапы
  - Окружения
  - Пример 1
  - Команды 1
  - Команды 2
  - Выражение 1
  - Выражение 2
  - Выражение 3
  - Координаты 1**
  - Координаты 2
  - Пример 2
  - Таблица 1
  - Таблица 2
  - Таблица 3
  - Таблица 4
- Графики 2

**coordinates** {⟨список\_координат⟩} (входные данные)

Определяет входные данные в виде явного списка координат в следующем формате:

- ✓ Координаты разделяются пробелами.
- ✓ Каждая координата представляет список значений через запятые, ограниченный круглыми скобками.
- ✓ Значения кроме чисел могут содержать любые выражения, поддерживаемые математическим модулем TikZ. Если выражение само содержит круглые скобки, его необходимо заключить в фигурные скобки.

# Определения источника данных из списка (окончание)

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Обзор

Этапы

Окружения

Пример 1

Команды 1

Команды 2

Выражение 1

Выражение 2

Выражение 3

Координаты 1

**Координаты 2**

Пример 2

Таблица 1

Таблица 2

Таблица 3

Таблица 4

Графики 2

- ✓ После координаты может следовать другая, отделяемая от первой символами «+-». В этом случае вторая координата определяет значения погрешностей для первой, которые отображаются на графиках при включении соответствующей настройки.
- ✓ После координаты (и, возможно, её погрешности) могут быть указаны произвольные метаданные в квадратных скобках, связанные с этой координатой. Способ их отображения зависит от настроек, например, в виде цвета (см. далее).
- ✓ Пустые строки означают разрывы функции.

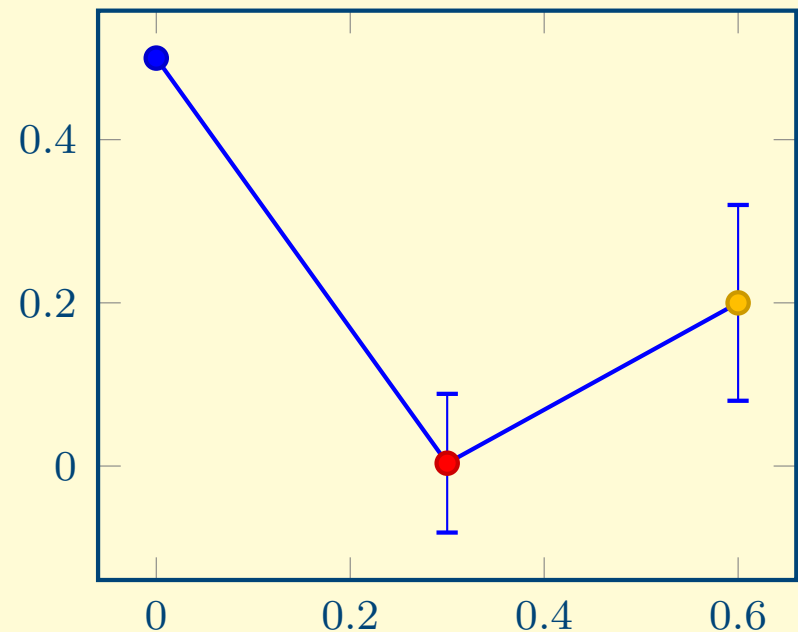
`math parser = true, false` (по умолчанию: **true**)

Включает/отключает синтаксический анализатор математических выражений для координат.

# Пример вывода графика по координатам

## Пример

```
\begin {axis} [width = 62mm]
  \addplot+ %
    [error bars/y dir = both,
    error bars/y explicit, %
    scatter, %
    point meta = explicit] %
    coordinates %
    { %
      (0, 0.5) [1] %
      (0.3, {sin(0.2)}) +- %
      (0.1, 0.085) [3] %
      (0.6, 0.2) +- %
      (1e-1, 0.12) [2] %
    };
\end {axis}
```



Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Обзор

Этапы

Окружения

Пример 1

Команды 1

Команды 2

Выражение 1

Выражение 2

Выражение 3

Координаты 1

Координаты 2

**Пример 2**

Таблица 1

Таблица 2

Таблица 3

Таблица 4

Графики 2

# Определения источника данных из таблицы

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1**
- Таблица 2
- Таблица 3
- Таблица 4
- Графики 2

**table** [*⟨выбор\_колонок⟩*] {⟨данные⟩} (входные данные)

Определяет входные данные в табличном виде из текстового файла с заданным именем через параметр *⟨данные⟩* или непосредственно передаваемые через этот параметр в следующем формате:

- ✓ По умолчанию строки разделяются переводами строки, ячейки — пробелами.
- ✓ Строки-комментарии (по умолчанию начинаются с «#» и «%») игнорируются.
- ✓ Первая строка-не-комментарий анализируется на содержание названий колонок. По этим именам можно в дальнейшем на них ссылаться. Иначе строка считается первой строкой числовых данных.

# Определения источника данных из таблицы (продолжение)

- ✓ Таблица может иметь несколько колонок. По умолчанию первая используется в качестве  $x$ -координат, вторая —  $y$ . При помощи параметров в  $\langle \text{выбор\_колонок} \rangle$  (см. далее) можно выбрать колонки для значений координат  $x$ ,  $y$ ,  $z$ , ошибок по каждой координате и метаданных.

**table/x** =  $\{ \langle \text{имя\_колонки} \rangle \}$

**table/x index** =  $\{ \langle \text{номер\_колонки} \rangle \}$

Позволяет указывать колонку для координаты  $x$  по имени или номеру (начиная с 0). Для этой и последующих настроек существуют аналогичные для координат  $y$ ,  $z$ , **x/y/z error (plus/minus)** и **meta**.

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2**
- Таблица 3
- Таблица 4
- Графики 2



# Определения источника данных из таблицы (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
  - Обзор
  - Этапы
  - Окружения
  - Пример 1
  - Команды 1
  - Команды 2
  - Выражение 1
  - Выражение 2
  - Выражение 3
  - Координаты 1
  - Координаты 2
  - Пример 2
  - Таблица 1
  - Таблица 2
  - Таблица 3**
  - Таблица 4
- Графики 2

**table/x expr** = {⟨выражение⟩}

Позволяет использовать для значений координаты  $x$  математическое выражение вместо данных из какой-либо колонки. В выражении могут использоваться следующие макросы:

Таблица 4: макросы для использования в выражениях

Макрос	Возвращаемое значение
<b>\thisrow</b> {⟨имя⟩}	Значение колонки с заданным именем в текущей строке
<b>\thisrowno</b> {⟨№⟩}	Аналогично, для номера колонки
<b>\coordindex</b>	Номер текущей строки данных, начиная с 0

# Определения источника данных из таблицы (окончание)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Обзор
- Этапы
- Окружения
- Пример 1
- Команды 1
- Команды 2
- Выражение 1
- Выражение 2
- Выражение 3
- Координаты 1
- Координаты 2
- Пример 2
- Таблица 1
- Таблица 2
- Таблица 3
- Таблица 4**
- Графики 2

**table/row sep** = `newline, \\` (по умолчанию: **newline**)

Устанавливает в качестве разделителя строк таблицы символ перевода строки или «\\».

**table/col sep** = `tab, comma, semicolon, ...` (по ум.: **space**)

Устанавливает в качестве разделителя строк таблицы заданные символы.

**table/skip first n** = `{⟨количество⟩}` (по умолчанию: `0`)

Позволяет игнорировать первые строки файла в заданном количестве.

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

**Графики 2**

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

# Визуализация данных в виде графиков (окончание)

# Типы двумерных графиков

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

**Типы 1**

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

## **sharp plot**

Ломаная линия между вершинами (по умолчанию).

## **smooth**

Плавная интерполированная линия.

## **const plot mark left**

Соединяет вершины последовательностью горизонтальных и вертикальных отрезков (соединяемые точки расположены на левых краях «ступеней»). Синоним: **const plot**.

## **const plot mark right**

Аналогично, на правых краях.

# Типы двумерных графиков (продолжение)

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

**const plot mark mid**

Аналогично, посередине.

**jump mark left**

**jump mark right**

**jump mark mid**

Аналогично, но без вертикальных отрезков.

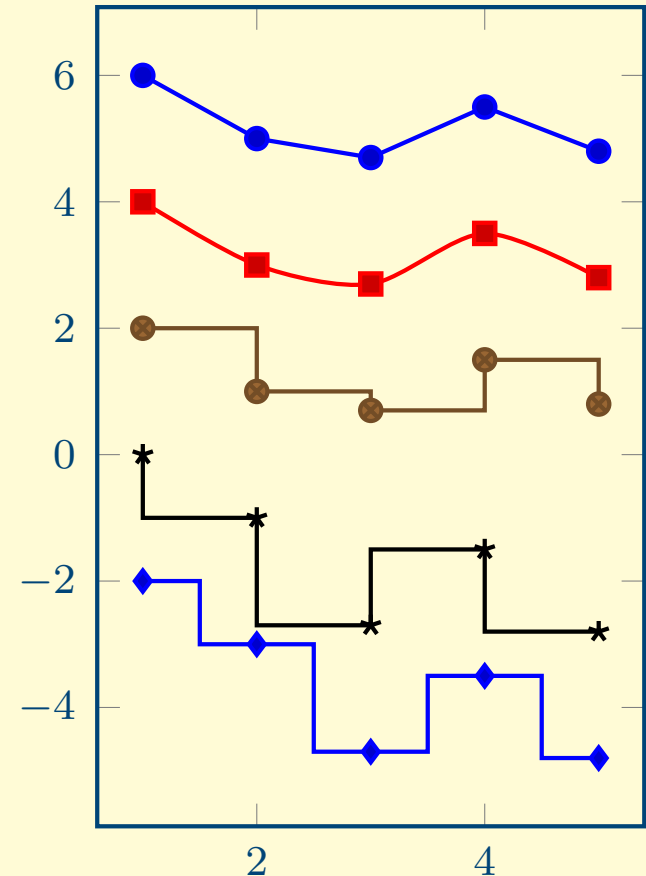
**only marks**

Рисует только отметки в координатах.

# Примеры типов графиков

## Пример

```
\begin {axis} %  
  [width = 52mm, height = 70mm]  
  \addplot+ table {data1.txt};  
  \addplot+ [smooth] %  
    table {data2.txt};  
  \addplot+ [const plot mark left] %  
    table {data3.txt};  
  \addplot+ [const plot mark right] %  
    table {data4.txt};  
  \addplot+ [const plot mark mid] %  
    table {data5.txt};  
\end {axis}
```



Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

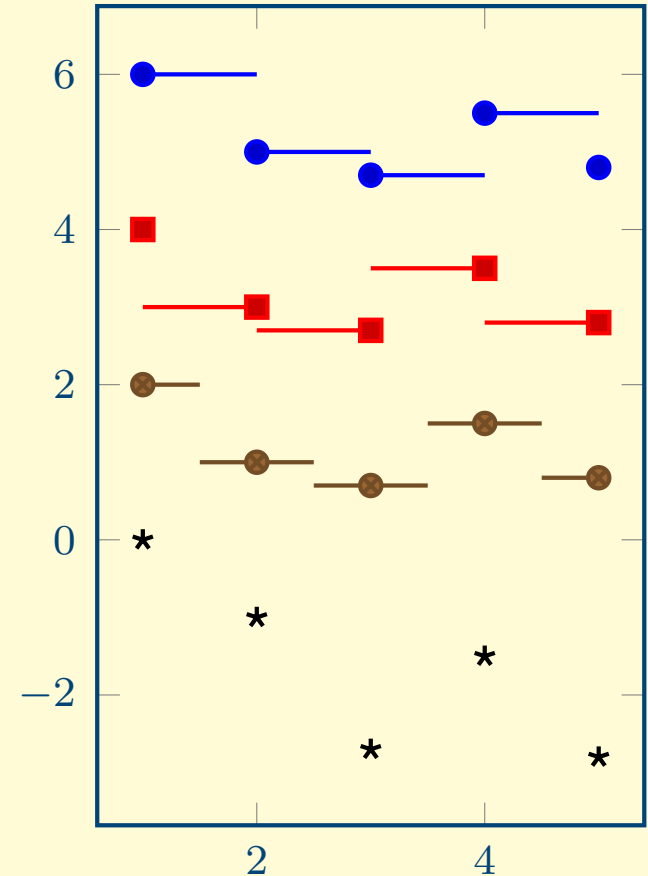
Пример 4

# Примеры типов графиков (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2**
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи
- Оси 1
- Оси 2
- Оси 3
- Оси 4
- Оси 4
- Пример 4

## Пример

```
\begin {axis} %  
  [width = 52mm, height = 70mm]  
  \addplot+ [jump mark left] %  
    table {data1.txt};  
  \addplot+ [jump mark right] %  
    table {data2.txt};  
  \addplot+ [jump mark mid] %  
    table {data3.txt};  
  \addplot+ [only marks] %  
    table {data4.txt};  
\end {axis}
```



# Типы двумерных графиков (продолжение)

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

## **ybar interval**

Рисует прямоугольники от оси  $\overrightarrow{OX}$  вверх между  $x$ -координатами соседних точек и высоты, определяемой  $y$ -координатой левой точки (таким образом, значение  $y$  последней точки не используется).

## **xbar interval**

Аналогично **ybar interval**, прямоугольники рисуются от оси  $\overrightarrow{OY}$  вправо.

## **ybar**

Столбики между осью  $\overrightarrow{OX}$  и заданными координатами.

## **xbar**

Аналогично, между осью  $\overrightarrow{OY}$  и заданными координатами.



# Примеры типов графиков (продолжение)

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

**Пример 3**

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

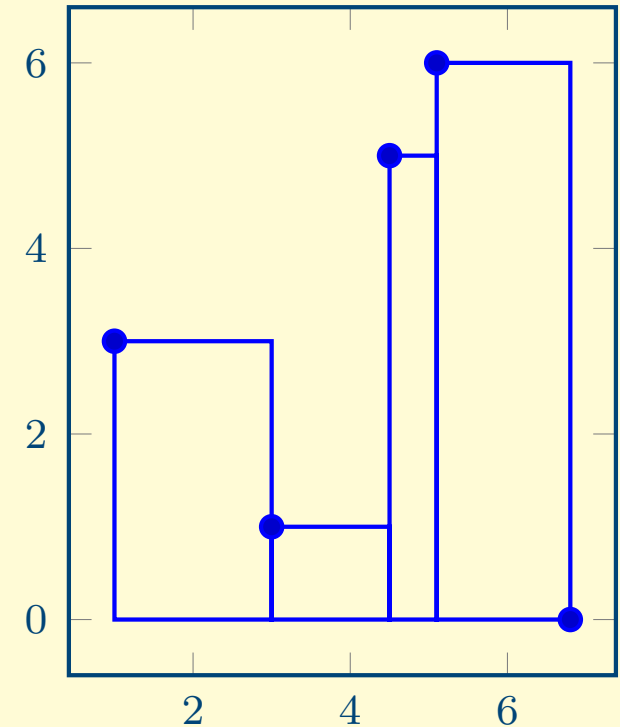
Оси 4

Оси 4

Пример 4

## Пример

```
\begin {axis} %  
  [width = 52mm, height = 60mm]  
  \addplot+ [ybar interval] %  
    table {data6.txt};  
\end {axis}
```



# Типы двумерных графиков (окончание)

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

## **scatter**

То же, что и `only marks`, но дополнительно меняет внешний вид отметок графика (по умолчанию цвет) в зависимости от метаданных, привязанных к координатам. Может быть также использован совместно с другими типами графиков. Источник метаданных и влияние на внешний вид определяются настройками, которые приведены ниже.

**scatter src** = none, *⟨выражение⟩*, ... (по ум.: none)

Определяет источник метаданных (см. табл. 5, 6). Является синонимом **point meta**. Ранее приводился пример использования ключа.

# Использование метаданных

Таблица 5: возможные значения ключа `scatter src`

Значение	Смысл
<b>none</b>	Отключает использование метаданных.
<b>x, y, z</b>	Использует соответствующую координату в качестве метаданных.
<b>f(x)</b>	Использует координату <b>y</b> при выводе двумерного графика и <b>z</b> для трёхмерного.
<b>explicit</b>	Указывает, что метаданные должны задаваться в явном виде вместе с координатами. Их формат подробно описан вместе с командами <code>\plot coordinates</code> и <code>\plot table</code> .

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

# Использование метаданных (окончание)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2**
- Подписи
- Оси 1
- Оси 2
- Оси 3
- Оси 4
- Оси 4
- Пример 4

Таблица 6: возможные значения ключа `scatter src` (окончание)

Значение	Смысл
<b>explicit symbolic</b>	Аналогично <b>explicit</b> , но метаданные представляются в виде произвольного текста. Используется при некоторых настройках отображения, например, для вывода метаданных возле отметок (см. <code>nodes near coords</code> ).
<i>⟨выражение⟩</i>	Определяет метаданные как результат вычисления выражения PGF. Может использовать переменные <code>x</code> , <code>y</code> , <code>z</code> , макросы табл. 4 и т. д.

# Вывод подписей возле отметок на графиках

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи**
- Оси 1
- Оси 2
- Оси 3
- Оси 4
- Оси 4
- Пример 4

**nodes near coords\*** = {*⟨содержимое⟩*} (по ум.: см. ниже)

Модификация настройки `scatter`, которая размещает вершины со значениями метаданных возле отметок графиков. *⟨содержимое⟩* определяет способ вывода, по умолчанию оно равно `\pgfmathprintnumber {\pgfplotspointmeta}` (вывод метаданных в виде числа). Если настройка `point meta` не указана, по умолчанию считается, что задано значение  $f(x)$ . Вариант «со звёздочкой» используется для комбинирования стиля с уже установленными.

**nodes near coords style** = {*⟨настройки\_стиля⟩*}

Добавляет настройки к стилю вершин для вывода метаданных.

# Настройки осей и графиков

Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

**Оси 1**

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4

**xtick** = `\empty`, `data`,  $\langle \text{список\_значений} \rangle$  (по умолчанию: `\{\}`)

Определяет расположение отметок вдоль оси  $\overrightarrow{OX}$ . Пустой список (по умолчанию) означает автоматическое расположение, `\empty` подавляет их использование,  $\langle \text{список\_значений} \rangle$  задаёт явно  $x$ -координаты (в формате команды PGF `\foreach`, включая возможное многоточие). Аналогично, определены ключи **ytick** и **ztick**.

**xtick distance** =  $\langle \text{число\_или\_пусто} \rangle$  (по умолчанию: пусто)

Определяет расстояние между отметками при их автоматическом расположении. Пустое значение означает автоматический выбор. Аналогично, **ytick distance** и **ztick distance**.

# Настройки осей и графиков (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи
- Оси 1
- Оси 2**
- Оси 3
- Оси 4
- Оси 4
- Пример 4

**xticklabels** =  $\langle \text{список\_меток} \rangle$  (по умолчанию:  $\{\}$ )

Определяет обозначения возле отметок вдоль оси  $\overrightarrow{OX}$ .

Аналогично, **yticklabels** и **zticklabels**.

**x tick label as interval** = true, false (по ум.: false)

Устанавливает обозначения между соседними отметками в виде интервалов. Аналогично, **y tick label as interval** и **z tick label as interval**.

**tick style** =  $\langle \text{стиль\_TikZ} \rangle$

Используется для рисования отметок в дополнение к **every tick**.

# Настройки осей и графиков (продолжение)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи
- Оси 1
- Оси 2
- Оси 3**
- Оси 4
- Оси 4
- Пример 4

**symbolic x coords** =  $\langle \text{список\_строк} \rangle$

Определяет символические обозначения для координат. Каждая строка сопоставляется с очередным целым числом, начиная с 0. Эти значения можно затем использовать во входных данных вместо чисел. Подход аналогичен использованию `xticklabels`, только проще (но менее гибко). Аналогично, **symbolic y coords** и **symbolic z coords**.

**enlarge x limits** = {auto, true, false, ...} (по ум.: auto)

Позволяет увеличивать размеры оси  $\overrightarrow{OX}$  по отношению к ширине графика. Значения в списке указываются из табл. 7, 8. Аналогично, **enlarge y limits** и **enlarge z limits**.



# Настройки осей и графиков (окончание)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи
- Оси 1
- Оси 2
- Оси 3
- Оси 4**
- Оси 4
- Пример 4

Таблица 7: возможные значения ключа `enlarge x limits`

Значение	Смысл
<b>true</b>	Увеличивает левую и правую границы.
<b>false</b>	Использует плотно прилегающие к графику границы.
<b>lower</b>	Увеличивает левую границу, правая прилегает плотно.
<b>upper</b>	Увеличивает правую границу.
<b>auto</b>	Увеличивает границы только если они были автоматически определены (для трёхмерных графиков работает только для оси $\vec{OZ}$ ).

# Настройки осей и графиков (окончание)

- Стили
- Деревья
- Вычисления 1
- Вычисления 2
- Графики 1
- Графики 2
- Типы 1
- Типы 2
- Пример 1
- Пример 2
- Типы 3
- Пример 3
- Типы 4
- Метаданные 1
- Метаданные 2
- Подписи
- Оси 1
- Оси 2
- Оси 3
- Оси 4
- Оси 4**
- Пример 4

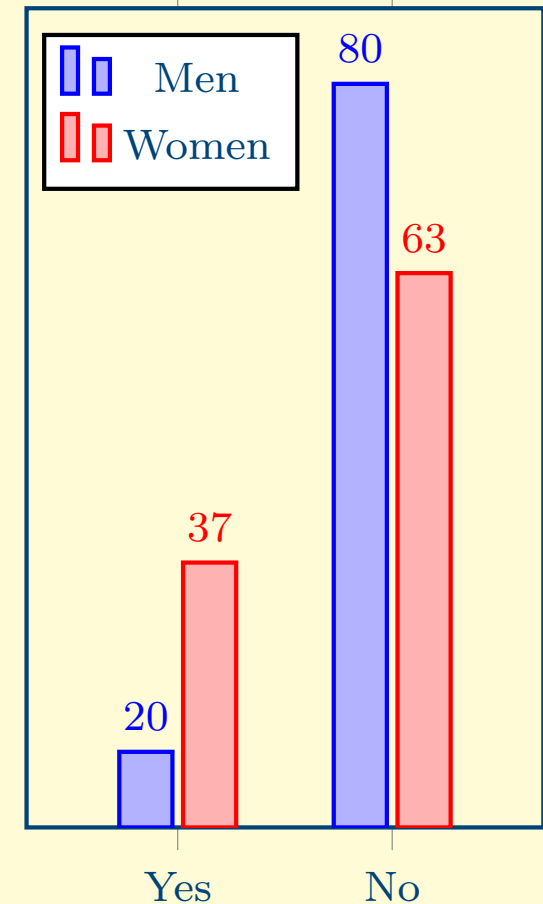
Таблица 8: возможные значения ключа `enlarge x limits`

Значение	Смысл
<code>value</code> =⟨коэф.⟩	Определяет коэффициент, на который умножается размер оси при увеличении. При отсутствии других значений возможна сокращённая запись: <code>enlarge x limits = ⟨коэф.⟩</code>
<code>abs value</code> =⟨коэф.⟩	Определяет коэффициент, на который умножается размер оси при увеличении.

# Пример столбиковой диаграммы

## Пример

```
\begin {axis} %  
  [width = 50mm, height = 70mm, %  
  ybar, nodes near coords, %  
  symbolic x coords = {Yes, No}, %  
  enlarge x limits = {abs = 10mm}, %  
  enlarge y limits = {abs = 5mm}, %  
  xtick = data, ytick = \empty, %  
  legend pos = north west]  
  \addplot+ coordinates %  
    {(Yes, 20) (No, 80)};  
  \addplot+ coordinates %  
    {(Yes, 37) (No, 63)};  
  \legend {Men,Women};  
\end {axis}
```



Стили

Деревья

Вычисления 1

Вычисления 2

Графики 1

Графики 2

Типы 1

Типы 2

Пример 1

Пример 2

Типы 3

Пример 3

Типы 4

Метаданные 1

Метаданные 2

Подписи

Оси 1

Оси 2

Оси 3

Оси 4

Оси 4

Пример 4