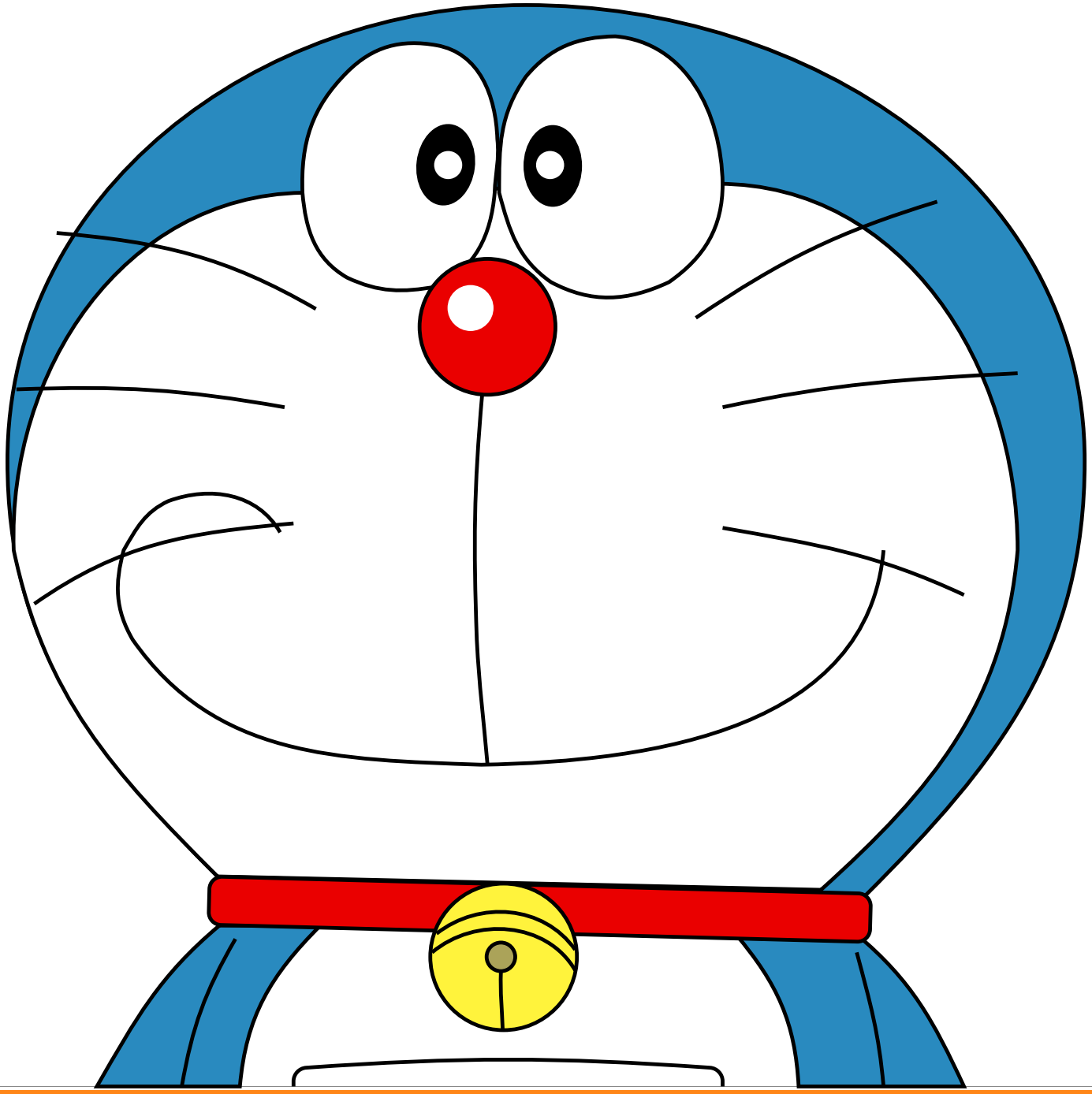


TikZ 学习入门之葵花宝典

未完续

作者：八一

Email： hoganbin1995@outlook.com

微信公众号：八一考研数学竞赛

完成时间： May 9, 2020

版本：3.09

ElegantLaTeX Program

人生的意义，在于折腾

目　　录

前言

在学习 LaTeX 与 TikZ 过程中，八一发现到大家其学习 LaTeX 曲线陡峭，它并非所见即所得，可能一个微小的改动需要长时间的全文编译，而且代码并不能真正显示事物本来的样子，TikZ 的底层是 PGF(portable graphics format)，即它是 PGF 的前端。

它也并不是唯一可以在 LaTeX 中画出漂亮图的宏包，比如 xfig 就可以代替它。另外 pstricks 和 metapost 都可以完成 TikZ 的工作；在 PGF 中有很多优秀的并且可以独立于 PGF 之外运行的包。比如 pgfpages 宏包，它功能强大，可以合并多页，可以制作水印，可以将多页缩放到一个页面里；pgfplots 宏包可直接调用一个 axis 环境和进行一些简单的优化来绘制函数并通过数据点拟合函数和散点图等。

那我们开始学习 TikZ，首先得有个前提工作：这里的`tikz`宏包的加载是必须的，而`tikz`有两种使用方法：一种命令式用`\tikz`命令，是行内模式；一种环境式用`tikzpicture`环境命令包围起来，是行间模式。如下所示：

```
\tikz{\draw (1,0) -- (0,1) -- (-1,0) -- (0,-1) -- cycle;}
\begin{tikzpicture}
\draw (1,0) -- (0,1) -- (-1,0) -- (0,-1) -- cycle;
\end{tikzpicture}
```

行内模式对于注重内容的用户相对较少，基本不用，这种模式下直接调用而`\tikz`命令。但不管是注重华丽表现效果还是注重内容的用户，总是会有需求需要某一整张图片来表达某些内容，而`tikz`以及其他基于`tikz`的宏包在命令行绘图这个领域是不错的。下面将主要使用`tikzpicture`环境命令模式。

用`tikz`绘制某个单独的图片而不是一般的 A4 页面推荐使用 standalone 类。如下所示：

```
\documentclass[tikz,border=2pt]{standalone}
\begin{document}
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw[->] (-3,0) -- (3,0);
\draw[->] (0,-3) -- (0,3);
\draw (0,0) circle (1);
\end{tikzpicture}
\end{document}
```

第 1 章 基础准备

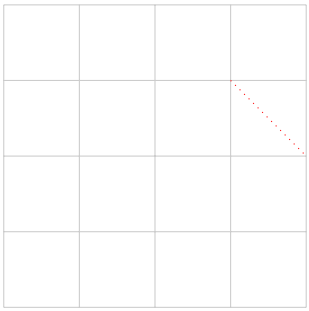
内容提要

- 点
- 圆
- 矩形
- 抛物线
- 文本
- 直线
- 抛物线
- pgfplots

1.1 点 (Point Path)

一行的中间高度画一个半径为一半行高的红点●使用 coordinate 命令或者 path 命令附带 coordinate 来定义一个点。

```
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\path (1,1) coordinate (p1);
\coordinate (p2) at ( 2, 0);
\draw[dotted, red] (p1) -- (p2) ;
\end{tikzpicture}
```



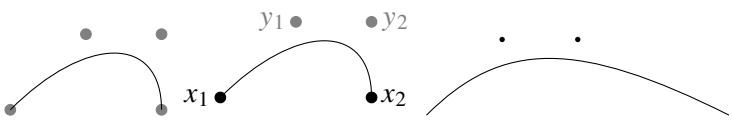
1.1.1 控制点

起点 x 的控制点 y，指的是曲线所在点 x 处的切线方向指向 y 点。如图所示。点 $x_1(0,0)$ 处的切线方向指向点 $y_1(1,1)$，点 $x_2(2,0)$ 点的切线方向指向点 $y_2(2,1)$，用法如下：

```
\draw[options] (x1,y1) .. controls (x2,y2) and (x3,y3) .. (x4,y4);
```

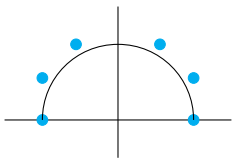
```
\begin{tikzpicture}
\filldraw[gray] (0,0) circle (2pt) (1,1) circle (2pt)
(2,1) circle (2pt) (2,0) circle (2pt);
```

```
\draw (0,0) .. controls (1,1) and (2,1) .. (2,0);
\end{tikzpicture}
\begin{tikzpicture}
\filldraw (0,0) circle [radius =2 pt ] node [left ] {$ x _1$};
\filldraw [gray ] (1,1) circle [radius =2 pt ] node [left ] {$ y _1$};
\filldraw [gray ] (2,1) circle [radius =2 pt ] node [right ] {$ y_2$};
\filldraw (2,0) circle [radius =2 pt ] node [right ] {$ x _2$};
\draw (0,0) .. controls (1,1) and (2,1) .. (2,0); % 核心代码
\end{tikzpicture}
\begin{tikzpicture}
\draw (0,0) .. controls (1,1) and (2,1) .. (4,0);
\fill (1,1) circle (1pt) (2,1) circle (1pt);
\end{tikzpicture}
```



再来一个控制点的例子：用控制点画一个半圆，见下图。当然，本例只是阐释控制点，实际中很少用这种方式画半圆。

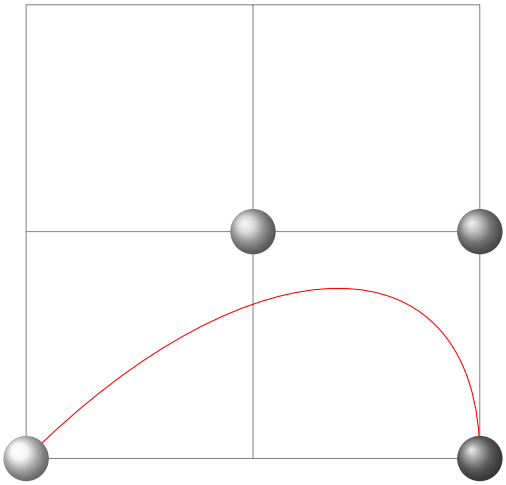
```
\begin{tikzpicture}
\draw (- 1.5,0) -- (1.5,0);
\draw (0,- 0.5) -- (0,1.5);
% show the control points
\filldraw [gray ] ( - 1,0) circle [radius =2 pt ]
(1,0) circle [radius =2 pt ]
[cyan ] (- 1,0.555) circle [radius =2 pt ]
[cyan ] (- 0.555,1) circle [radius =2 pt ]
[cyan ] (0.555,1) circle [radius =2 pt ]
[cyan ] (1,0.555) circle [radius =2 pt ];
\draw (- 1,0) .. controls (- 1,0.555) and (- 0.555,1) .. (0,1) .. controls
    (0.555,1) and (1,0.555) .. (1,0);
\end{tikzpicture}
```



贝塞尔曲线是四个点画出一个曲线，。其中第一个点是起点，第四个点终点，然后另外两个点是控制点。

```
\begin{tikzpicture}[scale=3]
```

```
\draw[help lines] (0,0) grid (2,2);
\draw[color=red] (0,0) .. controls (1,1) and (2,1) .. (2,0);
\shade[ball color=gray!10] (0,0) circle (0.1);
\shade[ball color=gray!40] (1,1) circle (0.1);
\shade[ball color=gray!70] (2,1) circle (0.1);
\shade[ball color=gray] (2,0) circle (0.1);
\end{tikzpicture}
```

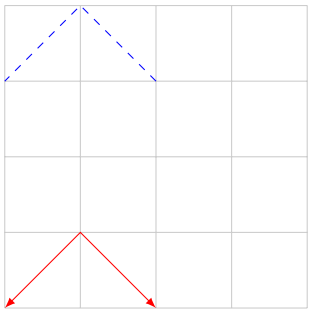


1.1.2 点的相对偏移

tikz 中有一个重要的概念，当前点，然后点可以通过当前点根据相对偏移来确定一个新的点。上面代码第 9 行的 ++ 符号和第 10 行的 + 符号都根据当前点然后进行了 Δx 和 Δy 的相对偏移从而确定了一个新的点。这两个符号的区别在于是不是更新当前点数据。++ 符号更新当前点，而 + 符号不更新。

++ 适合描述一连串逐渐变化的点，+ 适合描述多个点围绕着一个点变化的情况。

```
\begin{tikzpicture}[scale=1]
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw[latex-latex, red] (0,-2) -- ++(-1,1) -- ++(-1,-1);
\draw[dashed, blue] (0,1) -- +(-1,1) -- +(-2,0);
\end{tikzpicture}
```



1.1.3 node 命令中点的定义

tikz 中的点也支持极坐标表示，(30:1cm)，第一个参数是极座标里面的角度，第二个参数是半径。

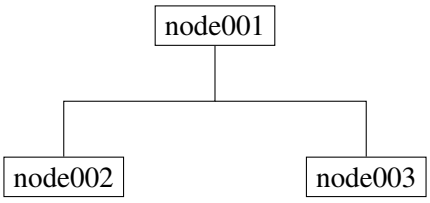
```
\begin{tikzpicture}
\node (node001) at (0,2) [draw] {test math competition};
\end{tikzpicture}
```

test math competition

从这里可以看到只要写上 draw 选项外面就会加上一个长方形，也就是 shape 的默认选项是 rectangle。如果你不希望外面有长方形，不写 draw 选项即可。

这里通过 node 命令定义了一个点，node001，在 (0,2) 那里。后面是可以使用的。

```
\begin{tikzpicture}
\node (node001) at (0,2) [draw] {node001};
\node (node002) at (-2,0) [draw] {node002};
\node (node003) at (2,0) [draw] {node003};
\draw (node cs:name=node003,anchor=north) |- (0,1);
\draw (node002.north) |- (0,1) -| (node cs:name=node001,anchor=south);
\end{tikzpicture}
```

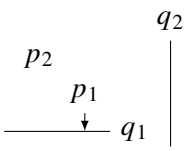


这里通过 node cs:name=node003 来获取之前那个 node 所在的点，然后通过 anchor=north 来定义那个 node 的接口在北边。除此之外的选项还有：south ，east ，west 。这里 |- 似乎是画垂直拐线的意思。上面的语法简写为可以 node002.north。

此外还有 angle 选项控制 node 接口的开口角度。

1.1.4 两个点定义出一个点

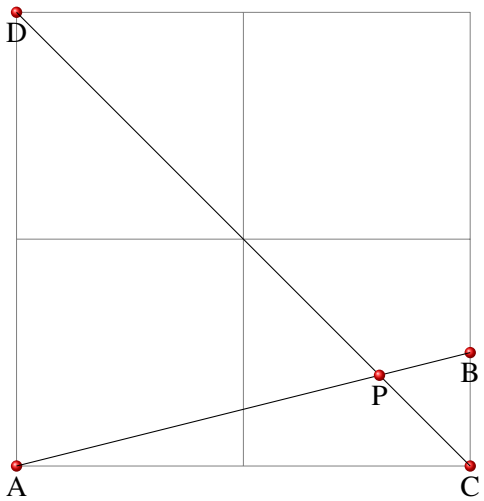
```
\begin{tikzpicture}
\node (p1) at (30:1) {$p_1$} ;
\node (p2) at (75:1) {$p_2$} ;
\draw (-0.2,0) -- (1.2,0) node[right] (xline) {$q_1$};
\draw (2,-0.2) -- (2,1.2) node[above] (yline) {$q_2$};
\draw[->] (p1) -- (p1 |- xline);
\end{tikzpicture}
```



这种形式 (p1 |- xline) 表示取第一个点的 x 和第二个点的 y 组成一个新的点。如果是 (p1 -| xline) 表示取第二个点的 x 和第一个点的 y 组成一个新的点。

两个 path 的交点

```
\begin{tikzpicture}[scale=3]
\draw[help lines] (0,0) grid (2,2);
\coordinate (A) at (0,0);
\coordinate (B) at (2,0.5);
\coordinate (C) at (2,0);
\coordinate (D) at (0,2);
\shade[ball color=red](A) circle (0.025) node[below] {A};
\shade[ball color=red](B) circle (0.025) node[below] {B};
\shade[ball color=red](C) circle (0.025) node[below] {C};
\shade[ball color=red](D) circle (0.025) node[below] {D};
\draw[name path=AB] (A) -- (B); \draw[name path=CD] (C) -- (D);
\path[name intersections={of=AB and CD}] (intersection-1) coordinate (P);
\shade[ball color=red](P) circle (0.025) node[below] {P};
\end{tikzpicture}
```



这个例子用到了点的定义，点的标出，以及 path 交点的定义，要用到 library：intersections 。有时候有些路径你不希望显示出来那么就用 path 命令来定义路径。

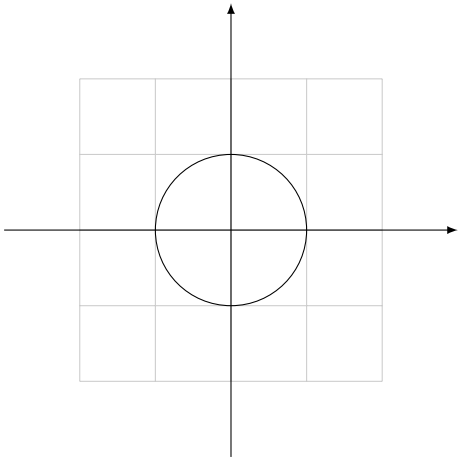
给新交点取名字：用 by 选项可以给画出来的交点取一个名字，默认的 intersection-1 之类的也可以使用。此外还可以加上选项：

```
\path[name intersections={of=D and E,by={[label=above:$C$]C,[label=below:$C'$]C'}}];
```

1.2 圆 (Circle Path)

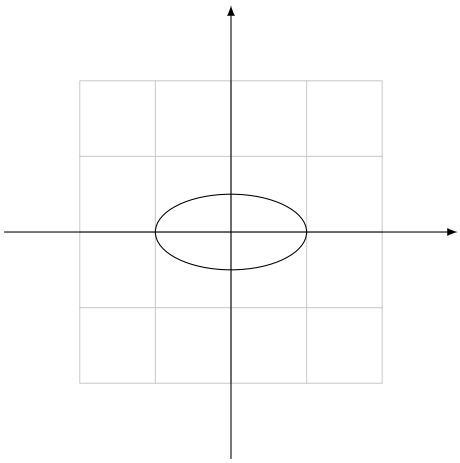
我们已经知道如何使用 tikz 在行内画图，下面我们用以下代码在文中画出图

```
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw[->] (-3,0) -- (3,0);
\draw[->] (0,-3) -- (0,3);
\draw (0,0) circle (1);
\end{tikzpicture}
```



其中第一个点是圆中心，`circle`表示画圆，第二个参数是半径大小.

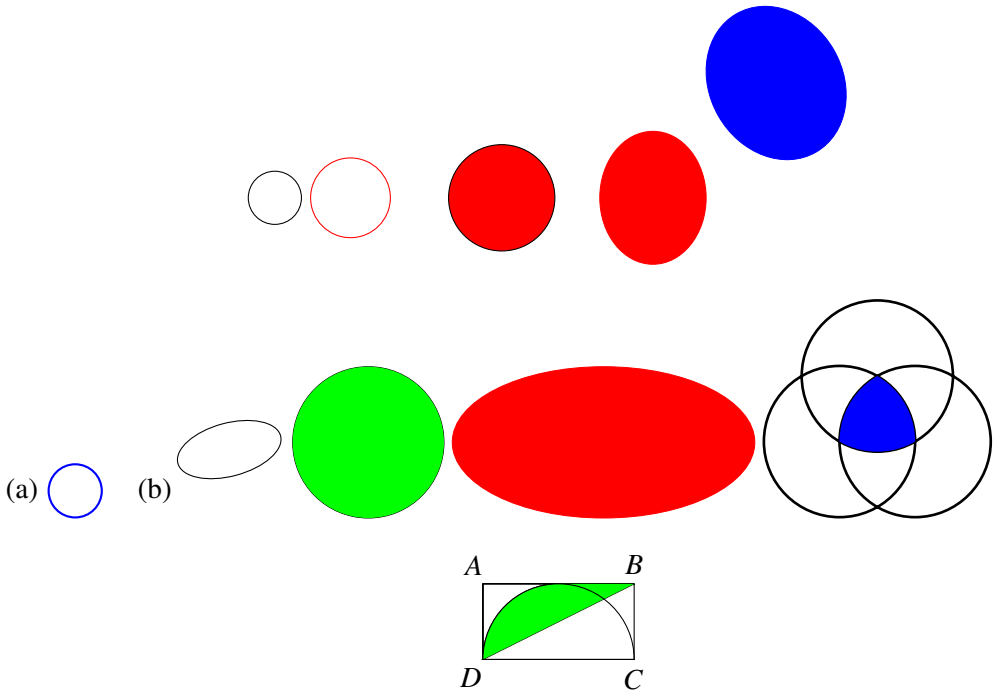
```
\begin{tikzpicture}
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw[->] (-3,0) -- (3,0);
\draw[->] (0,-3) -- (0,3);
\draw (0,0) ellipse (1 and 0.5);
\end{tikzpicture}
```



这里第一个点是椭圆的中心点，ellipse 表示画椭圆，后面参数两个值第一个是 a 也就是椭圆的半长轴，第二个是 b 也就是椭圆的半短轴。用法：

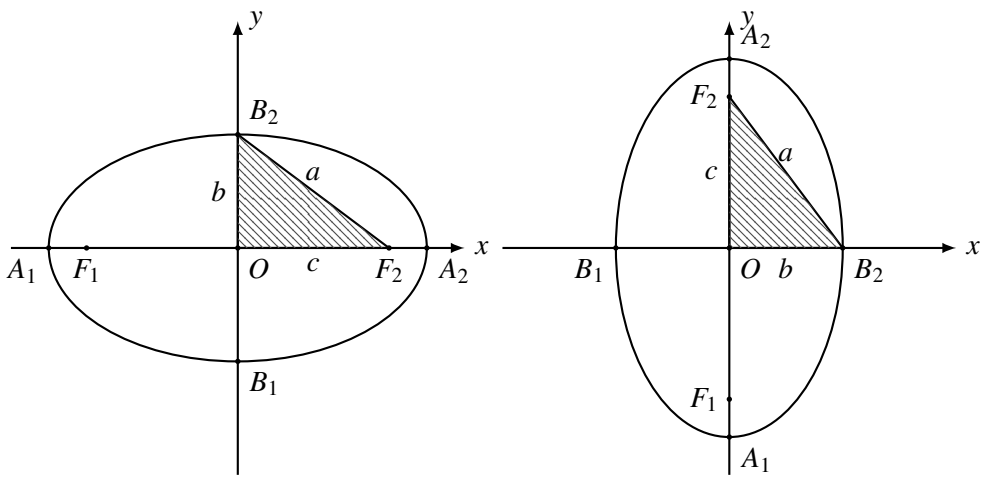
```
\draw[options] (x,y) circle (raidus);
\draw[options] (x,y) ellipse (x.raidus anda y.radius);
```

```
\begin{tikzpicture}
\draw (0,0) circle (10pt);
\draw[red] (1,0) circle (15pt);
\draw[fill=red] (2,0) circle (20pt);
\draw[red,fill=red] (3,0) ellipse (20pt and 25pt);
\filldraw[blue,rotate=30] (3.5,-2) ellipse (25pt and 30pt); % another way
\end{tikzpicture}

\begin{tikzpicture}
\draw (- 20pt,0) node [auto] {(a)};
\draw [thick ,blue ] (0pt ,0pt ) circle [radius =10 pt ] ;

\draw (30pt,0) node [auto] {(b)};
\draw [rotate =15] (60pt ,0) ellipse [x radius =20 pt , y radius =10 pt ];
\end{tikzpicture}
\tikz {\draw (0,0) circle (1);\fill [green](0,0) circle (1);}
\tikz \filldraw [red] (0,0) ellipse (2 and 1);
\begin{tikzpicture}[line width=1pt]
\draw (0,0)circle(1.0) (1,0)circle(1.0)
(60:1)circle(1.0);
\clip (0,0) circle (1.0);
\clip (1,0) circle (1.0);
\fill[blue] (60:1)circle(1.0);
\end{tikzpicture}
\begin{center}
\tikz {\draw (0,0)coordinate(D) node[below left=-0.5pt and -4pt]{$D$} --(0,1)
    coordinate(A) node[above left=-0.5pt and -4pt]{$A$}--(2,1)coordinate(B)
    node[above]{$B$} -- (1, 1) arc [start angle=90, end angle=180, radius=1]
    rectangle (2,1)--(0,0)-- (2, 0)coordinate(C) node[below]{$C$} ;
\fill [green](0, 0) -- (2, 1) -- (1, 1)arc [start angle=90, end angle=180,
    radius=1] -- cycle;
\clip (0,0) rectangle (2,1);
\draw (1,0) circle(1);
\draw (0,0) -- (2,0);}
\end{center}
```



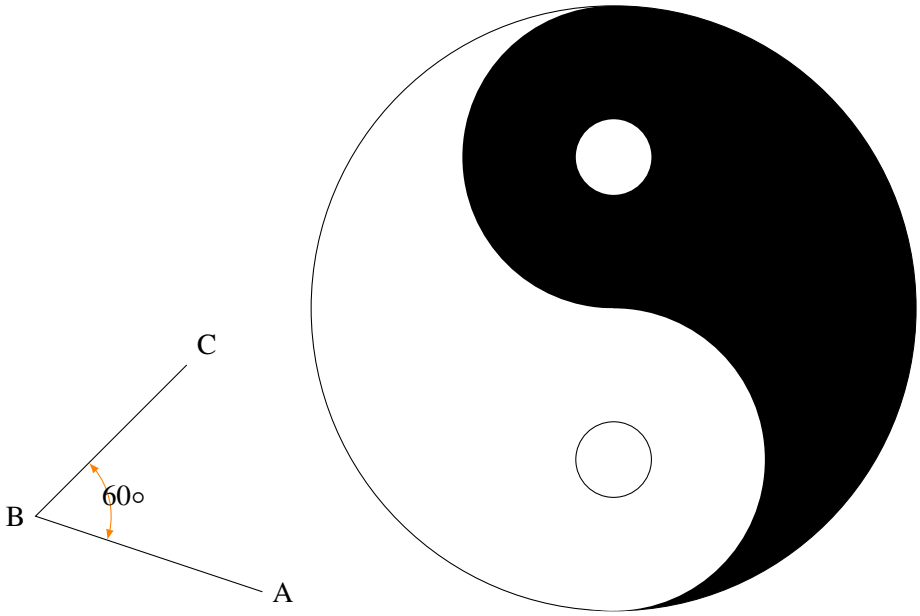

```
\begin{tikzpicture}[scale=0.5,thick]
\def\a{5}%长半轴
\def\b{3}%短半轴
\def\c{4}%焦半轴
\def\ptsize{2.0pt} %点的半径

%x 轴 和 y轴
\path[name path=xaxis,thick,draw,->](-6,0)--(6,0) node[right] {$x$};
\path[name path=yaxis,thick,draw,->](0,-6)--(0,6) node[above,right] {$y$};
%x 轴 与 y 轴的交点
\path [name intersections={of = xaxis and yaxis}];
\coordinate[label=below right:$O$] (O) at (intersection-1);
%画一个椭圆
\draw [name path = myellipse ] (intersection-1) ellipse (\a cm and \b cm);
%椭圆与x 轴交点
\path [name intersections={of = xaxis and myellipse}];
\coordinate[label=below right:$A_2$] (a2) at (intersection-1);
\coordinate[label=below left:$A_1$] (a1) at (intersection-2);
%椭圆与y 轴交点
\path [name intersections={of = yaxis and myellipse}];
\coordinate[label=above right:$B_2$] (b2) at (intersection-1);
\coordinate[label=below right:$B_1$] (b1) at (intersection-2);
%焦点
\coordinate[label=below :$F_1$] (f1) at (-\c,0);
\coordinate[label=below :$F_2$] (f2) at (\c,0);
%a b c 的几何意义
```

```
\draw (b2) --(f2) node[midway,above] {$a$};
\draw (b2) --(O) node[midway,left] {$b$};
\draw (O) --(f2) node[midway,below] {$c$};
%阴影部分填充
\fill [pattern =north west lines,pattern color = black!70](b2)--(O)--(f2)--
    cycle;
%画点
\foreach \p in {O,a1,a2,b1,b2,f1,f2}
\fill (\p) circle (\ptsize);
%在图像右侧再画一个焦点在y轴上的椭圆
\begin{scope}[xshift=13cm]
\path[name path=xaxis,thick,draw,->](-6,0)--(6,0) node[right] {$x$};
\path[name path=yaxis,thick,draw,->](0,-6)--(0,6) node[above,right] {$y$};
\path [name intersections={of = xaxis and yaxis}];
\coordinate[label=below right:$O$] (O) at (intersection-1);
%画一个椭圆
\draw [name path = myellipse ] (intersection-1) ellipse (\b cm and \a cm);

\path [name intersections={of = xaxis and myellipse}];
\coordinate[label=below right:$B_2$] (a2) at (intersection-1);
\coordinate[label=below left:$B_1$] (a1) at (intersection-2);

\path [name intersections={of = yaxis and myellipse}];
\coordinate[label=above right:$A_2$] (b2) at (intersection-1);
\coordinate[label=below right:$A_1$] (b1) at (intersection-2);

\coordinate[label=left :$F_1$] (f1) at (0,-\c);
\coordinate[label=left :$F_2$] (f2) at (0,\c);

\draw (a2) --(f2) node[midway,above] {$a$};
\draw (a2) --(O) node[midway,below] {$b$};
\draw (O) --(f2) node[midway,left] {$c$};

\fill [pattern =north west lines,pattern color = black!70](a2)--(O)--(f2)--
    cycle;

\foreach \p in {O,a1,a2,b1,b2,f1,f2}
\fill (\p) circle (\ptsize);

\end{scope}
\end{tikzpicture}
```



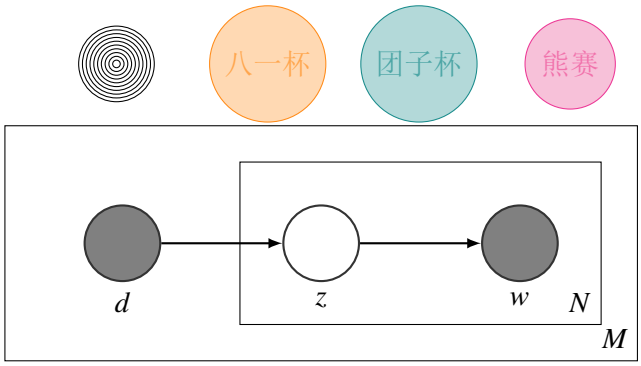

```
\begin{tikzpicture}
\draw(3,-1) coordinate (A) node[right] {A}
-- (0,0) coordinate (B) node[left] {B}
-- (2,2) coordinate (C) node[above right] {C}
pic["$60\circ$", draw=orange, <->, angle eccentricity=1.2, angle radius=1cm]
{angle=A--B--C};
\end{tikzpicture}

\begin{tikzpicture}
\draw(0,0) circle(4cm); %画一个4cm的圆圈
%把圆的右半边填黑
\begin{scope}
\clip(0,0) circle(4cm);
\fill[black] (0,-4) rectangle (4,4);
\end{scope}
%填黑八卦图左边的半圆
\begin{scope}
\clip(0,2) circle(2cm);
\fill[black] (-4,0) rectangle (4,4);
\end{scope}
%填白八卦图右边的半圆
\begin{scope}
\clip(0,-2) circle(2cm);
\fill[white] (-4,0) rectangle (4,-4);
\end{scope}
%把黑色部分的小圆圈填为白色
\begin{scope}
\clip(0,2) circle(0.5cm);
\fill[white] (-4,0) rectangle (4,4);
\end{scope}
%绘制下面的白色圆圈
```

```
\draw(0,-2) circle(0.5cm);
\end{tikzpicture}
```




```
\begin{center}
\begin{tikzpicture}
\foreach \x in {0,1,...,10} {
\node[draw, circle, inner sep=0pt, minimum size=\x mm] {};
}
\foreach \a/\x/\y in {1/orange/八一杯, 2/teal/团子杯, 3/magenta/熊赛} {
\node[circle, minimum size=2*\a mm, draw=\x!90, fill=\x!30, text=\x!70] at (2*
    \a, 0) {\y};
}
\end{tikzpicture}

\begin{tikzpicture}
\tikzstyle{main}=[circle, minimum size = 10mm, thick, draw =black!80, node
    distance = 16mm]
\tikzstyle{connect}=[-latex, thick]
\tikzstyle{box}=[rectangle, draw=black!100]
\node[main, fill = white!100] (z) [label=below:\emph{z}] { };
\node[main, fill = black!50] (d) [left=of z,label=below:$d$] { };
\node[main, fill = black!50] (w) [right=of z,label=below:$w$] {};
\path (d) edge [connect] (z)
(z) edge [connect] (w);
\node[rectangle, inner sep=0mm, fit= (z) (w),label=below right:$N$, xshift=13mm
    ] {};
\node[rectangle, inner sep=5.6mm,draw=black!100, fit= (z) (w)] {};
\node[rectangle, inner sep=4.8mm, fit= (z) (w),label=below right:$M$, xshift
    =12.5mm] {};
```

```
\node[rectangle, inner sep=10.4mm, draw=black!100, fit = (d) (z) (w)] {};
\end{tikzpicture}
\end{center}
```

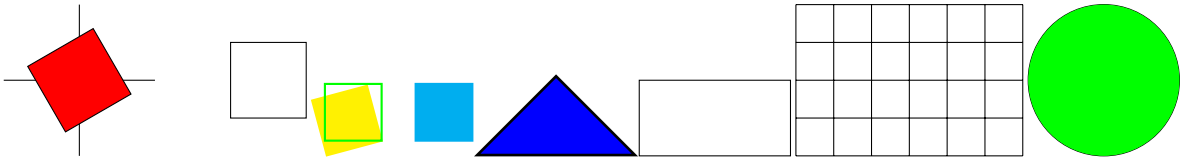


1.3 矩形与多边形 (Rectangle Path)

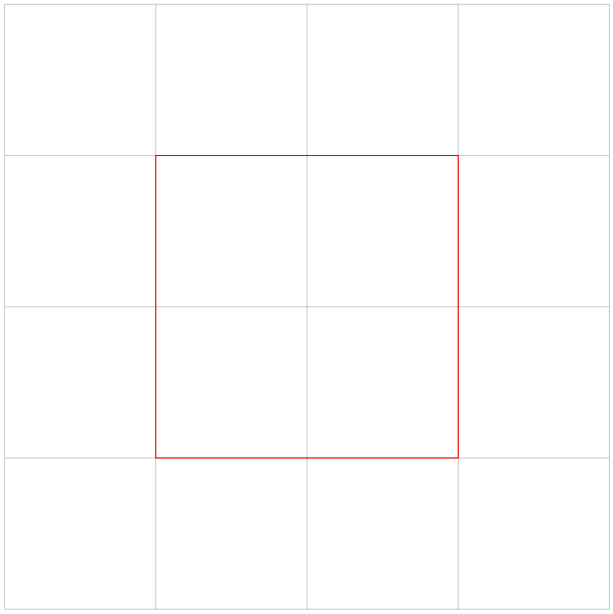
画正方形或矩形使用关键字 rectangle ，左下角和右上角坐标控制整个矩形的形状。如果将矩形旋转，则旋转角度以右下角坐标为轴心，逆时针方向旋转。用法如下：

```
\draw[options] (x1,y1) rectangle (x2,y2);
\draw[options] (x,y) rectangle +(width,height);
```

```
\begin{tikzpicture}
\draw (-1,0) -- (1,0);
\draw (0,-1) -- (0,1);
\draw[rotate=30, fill=red] (-0.5,-0.5) rectangle (0.5,0.5);
\draw (2,-0.5) rectangle +(1,1);
\end{tikzpicture}
\begin{tikzpicture}[scale=1.5]
% 第一个坐标为左下角，第二个坐标为右上角。
\filldraw [thick ,cyan ] (0.3,0) rectangle (0.8, 0.5);
% 旋转角度以右下角坐标为轴心 ,逆时针旋转。
\filldraw [thick ,yellow ,rotate =15] (- 0.5,0) rectangle (0,0.5);
\draw [thick ,green ] (- 0.5,0) rectangle (0,0.5);
\end{tikzpicture}
\begin{tikzpicture}[line width=2pt]
\draw (0,0) --(1,1) --(2,0) --cycle;
\fill[blue] (0,0) --(1,1) --(2,0) --cycle;
\end{tikzpicture}
\tikz \draw (0,0) rectangle (2,1);
\tikz \draw [step=0.5] (0,0) grid (3,2);
\tikz {\draw (0,0) circle (1);\fill [green](0,0) circle (1);}
```



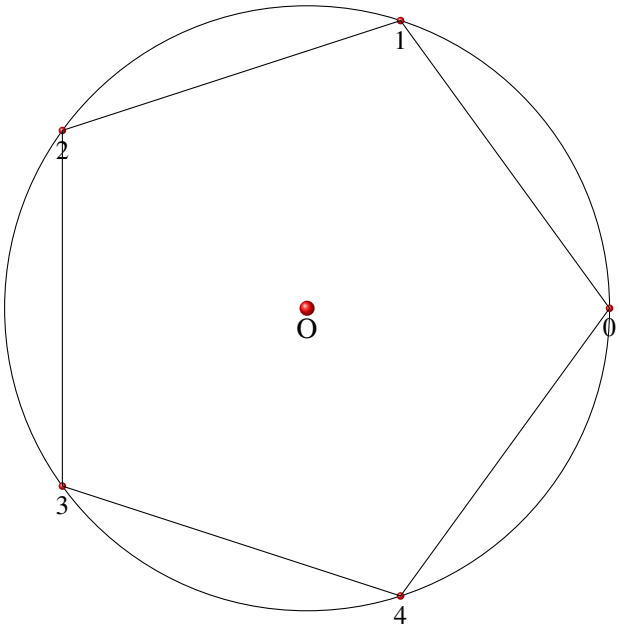
```
\begin{tikzpicture}[scale=2]
\draw[step=1,color=gray!40] (-2,-2) grid (2,2);
\draw[color=red] (-1,-1) rectangle (1,1);
\end{tikzpicture}
```



这里使用了可选项 color=red 来控制线条的颜色，然后画长方形的第一个点是左底点，rectangle 表示画长方形，第二个点表示右顶点。如果想要放大图形，可在 tikzpicture 环境后面跟上可选项 [scale=2]，即将图形放大两倍。

```
\begin{tikzpicture}
\draw (0,0) circle (4) ;
\coordinate (O) at (0,0);
\shade[ball color=red](O) circle (0.1) node[below] {O};
\def\n{5}
\pgfmathsetmacro\i{\n-1}
\foreach \x in {0,...,\i}
{
  \def\pointname{\x}
  \coordinate (\pointname) at ($(0,0) +(\x*360/\n:4cm)$) ;
  \shade[ball color=red](\pointname) circle (0.05) node[below] {\small \x};
}

\draw (O)
\foreach \x in {0,...,\i}
```

```
{ -- (\x) } -- cycle;

\end{tikzpicture}
```



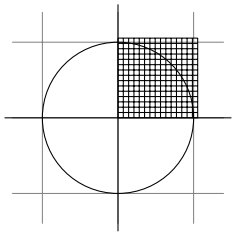
这个例子核心内容是批量定义点和点的运算，把这个弄懂了，后面 tikz 的核心大门就为你打开了，然后很多图形都可以用简洁的命令生成出来了。

1.4 网格 (Grid Path)

网格主要用于辅助绘图，其中 help lines 是个不错的参数设置。其用法：

```
\draw[options] (x1,y1) grid (x2,y2);
```

```
\begin{tikzpicture}
\draw[step=1.0cm, gray, very thin] (-1.4,-1.4) grid (1.4,1.4);
\draw (-1.5,0) -- (1.5,0);
\draw (0,-1.5) -- (0,1.5);
\draw (0,0) circle (1cm);
\draw[step=2pt] (0,0) grid (30pt,30pt);
\end{tikzpicture}
```



此外 step 用来控制网格之间的间距，可以 color 来设置网格的颜色，不过一般没那个必要。然后接下来第一个坐标点是网格的左底点，第二个坐标点是网格的右定点。

```
\begin{tikzpicture}
\draw[help lines] ( -5,-5 ) grid ( 5, 5);
\end{tikzpicture}
```

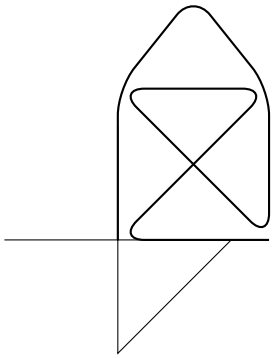
我们看到 tikz 的每一条命令最后都要跟一个分号";"。

1.5 直线 (Straight Path)

直线就是两个坐标点相连，中间 – 符号表示直线的意思。用法如下：

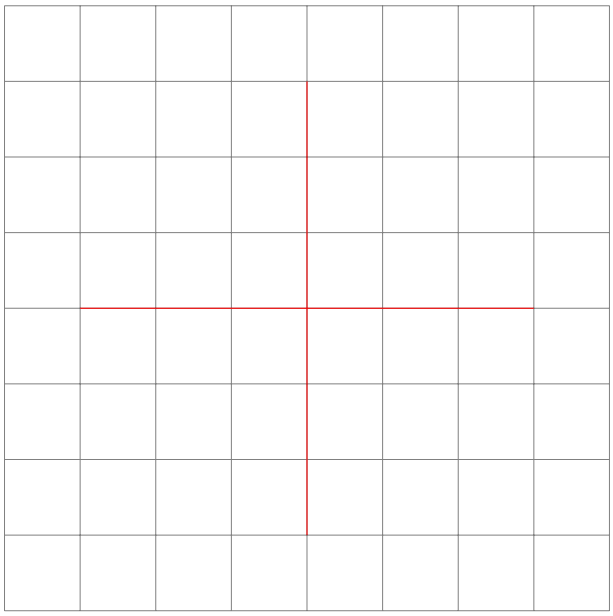
```
\draw[options] (x1,y1) -- (x2,y2) -- (x3,y3);
```

```
\begin{tikzpicture}
\draw (-1.5,0) -- (1.5,0) -- (0,-1.5) -- (0,1.5);
\draw[thick, rounded corners=10pt] (0,0) -- (0,2) -- (1,3.25) --
(2,2) -- (2,0) -- (0,2) -- (2,2) -- (0,0) -- (2,0);
\end{tikzpicture}
```



之前网格是 grid 表示网格的意思。如果几个点用 – 符号连接起来，表示这几个点连着来画几条折线，有多个画直线命令依次执行的意思。

```
\begin{tikzpicture}
\draw[help lines] ( -4,-4 ) grid (4,4);
\draw[red] (-3,0) -- (3,0);
\draw[red] (0,-3) -- (0,3);
\end{tikzpicture}
```



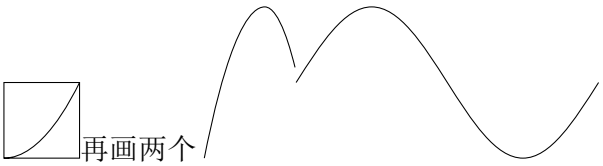
如果在直线带上箭头，draw 命令可以跟上可选项 `->`，这样直线的右端就有一个箭头了。此外还有: `->>`,`->|` , `-to`,`-latex` ,`-stealth`，他们的效果从上到下依次演示如下：

```
\begin{tikzpicture}
\draw[->] (-3,3) -- (3,3);
\draw[->>] (-3,2) -- (3,2);
\draw[->|] (-3,1) -- (3,1);
\draw[-to] (-3,0) -- (3,0);
\draw[-latex] (-3,-1) -- (3,-1);
\draw[-stealth] (-3,-2) -- (3,-2);
\end{tikzpicture}
```

1.6 抛物线 (Parabola Path)

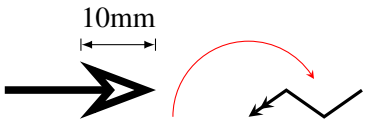
画实例用的抛物线使用关键字 parabola，用法如下：

```
\draw[options] (x1,y1) parabola (x2,y2);
```



再画两个

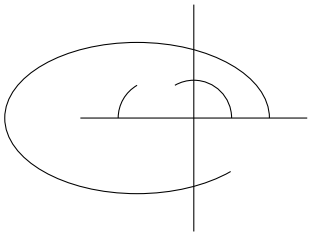
第一个点 (0,0) 是起点，使用 sin 画到第二点 (1,1) 。如果没有给出起点，那么 cos 会以 (1,1) 为起点，画到点 (2,0)。[x=1ex,y=1ex] 是表示在一个行距高度内，宽度为 1.57 倍的行距内画图。section 箭头 Tikz 默认的箭头有点难看，主要是因为它指的不精确。一个名字叫作“悄悄地 (stealth) ”的箭头。



以及 Arc Path 的使用来绘制曲线，用法：

```
\draw (x,y) arc (angle1:angle2:radius);
\draw (x,y) arc [start angle=angle1, end angle=angle2, radius=radius];
\draw (x,y) arc (angle1:angle2:x.radius and y.radius);
\draw (x,y) arc [start angle=angle1, end angle=angle2, x radius=rx, y radius=ry]
```

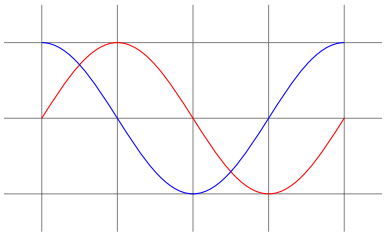
```
\begin{tikzpicture}
\draw (-1.5,0) -- (1.5,0);
\draw (0,-1.5) -- (0,1.5);
\draw (0.5,0) arc (0:120:0.5cm);
\draw (1,0) arc (0:315:1.75cm and 1cm);
\draw (-1,0) arc [start angle=180, end angle=120, radius=0.5cm];
% 以上不是推荐的方式
\end{tikzpicture}
```



正弦或余弦曲线使用关键字 sin/cos，用法：

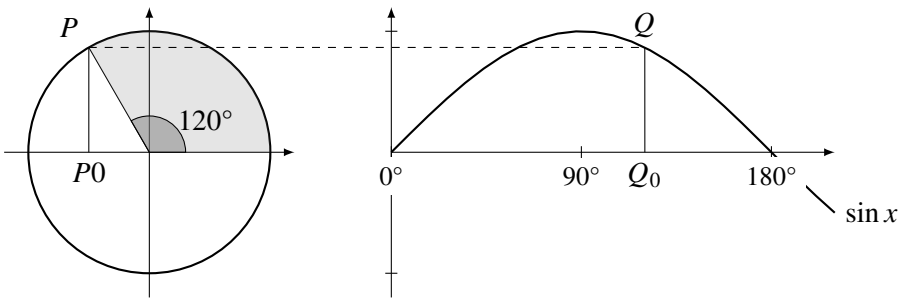
```
\draw[options] (x1,y1) sin (x2,y2);
\draw[options] (x1,y1) cos (x2,y2);
```

```
 \begin{tikzpicture}
\draw[help lines] (-0.5,-1.5) grid (4.5,1.5);
\draw[red] (0,0) sin (1,1) cos (2,0) sin (3,-1) cos (4,0);
\draw[blue] (0,1) cos (1,0) sin (2,-1) cos (3,0) sin (4,1);
\end{tikzpicture}
```



```
\begin{tikzpicture}[scale=1.6]%scale参数可以使得图形放大一定的倍数而本身的字体大
    小可以保持不变。
\def\iangle{120}
%画左边的圆
%scope环境里够成一整个区块，然后可以使这一整个区块进行平移。
\begin{scope}[xshift=-2cm]
\draw[->] (-1.2,0) --(1.2,0);
\draw[->] (0,-1.2) --(0,1.2);
\draw[thick] (0,0) circle(1cm);
\coordinate [label = \iangle:$P$] (P) at (\iangle:1);
\coordinate [label = below:$P0$] (P0) at (P |-0,0);
\draw (P)--(P0);
\draw (0,0)--(P);
\fill [fill = gray,fill opacity=0.2] (0,0)--(0:1) arc (0:\iangle:1)--cycle;
\filldraw [fill = gray,fill opacity=0.5] (0,0)--(0:0.3) arc (0:\iangle:0.3)--
    cycle;
\node [right] at (\iangle/2:0.3) {\ang{\iangle}};

\end{scope}
```

```
%画右边的正弦曲线
\draw[->] (0,0) --({rad(210)},0);
\draw[->] (0,-1.2) --(0,1.2);
\draw [thick,domain=0:rad(210)] plot (\x,{sin(\x r)}) node [right] {$\sin x$};

\foreach \t in {0,90,180}
{
\draw ({rad(\t)},-0.05)--({rad(\t)},0.05) ;
\node [below,outer sep =2pt ,font=\small , fill = white] at ({rad(\t)},0) {\ang
    {\t}};
}
\foreach \y in {-1,1}
{
\draw (-0.05,\y) -- (0.05,\y);
}
\coordinate [label=above:{$Q$}] (Q) at ({rad(\iangle)},{sin(\iangle)});
\coordinate [label=below:{$Q_0$}] (Q0) at (Q |- 0,0);
\draw (Q)--(Q0);
%左右相互连接
\draw[dashed] (P) --(Q);
\end{tikzpicture}
```



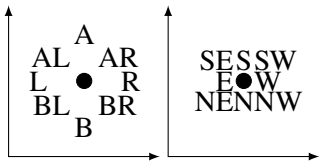
1.7 添加文本 (Add text)

添加文字比较简单，在已知位置上用 node 标出文字即可。其用法：

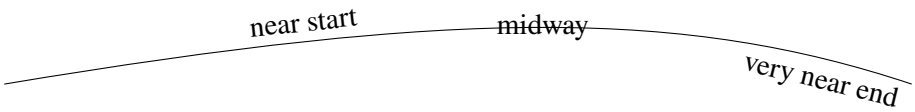
```
\draw (x,y) node[options] {text};
\draw (x,y) node[options] {text};
\node[options] at (x,y) {text};
options: above,below,left,right, or anchor=north,south,west,east.
```

```
\begin{tikzpicture}[scale=2]
\draw[<->] (0,1) -- (0,0) -- (1,0);
\draw[fill] (0.5,0.5) circle (0.05);
\draw (0.5,0.5) node[above=10pt] {A} node[left=10pt] {L}
```

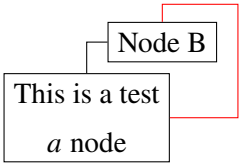
```
node[below=10pt] {B} node[right=10pt] {R};
\draw (0.5,0.5) node[above left=2pt] {AL} node[below left=2pt] {BL}
node[below right=2pt] {BR} node[above right=2pt] {AR};
\end{tikzpicture}
\begin{tikzpicture}[scale=2]
\draw[<->] (0,1) -- (0,0) -- (1,0);
\draw[fill] (0.5,0.5) circle (0.05);
\draw (0.5,0.5) node[anchor=north] {N} node[anchor=west] {W}
node[anchor=south] {S} node[anchor=east] {E};
\draw (0.5,0.5) node[anchor=north west] {NW} node[anchor=south west] {SW}
node[anchor=south east] {SE} node[anchor=north east] {NE};
\end{tikzpicture}
```



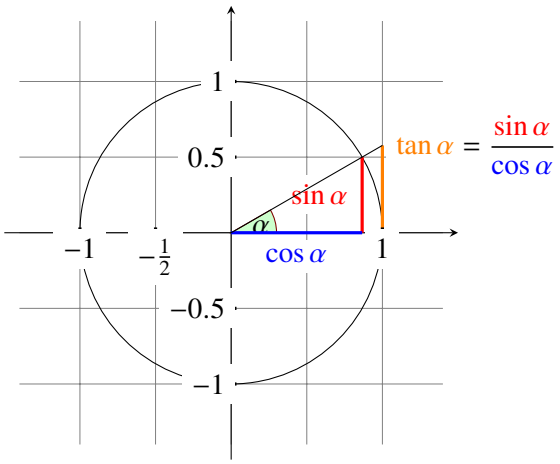

```
\begin{tikzpicture}
\draw (0,0) .. controls (6,1) and (9,1) ..
node[near start,sloped,above] {near start}
node {midway}
node[very near end,sloped,below] {very near end} (12,0);
\end{tikzpicture}
```




```
\begin{tikzpicture}
\draw (0,0) node(a) [draw,align=center] {This is a test\\$a$ node}
(1,1) node(b) [draw] {Node B};
\draw (a.north) |- (b.west);
\draw[color=red] (a.east) -| (2,1.5) -| (b.north);
\end{tikzpicture}
```



```
\begin{tikzpicture}[scale=2, >= stealth]
\draw[step=0.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
\filldraw[fill=green!20!white,draw=red!50!black] (0,0) -- (3mm,0mm)
arc [start angle=0, end angle=30, radius=3mm] -- cycle ;
\draw (2mm, 0.4mm) node {$\alpha$};
\draw[->] (-1.5,0) -- (1.5,0) coordinate (x axis);
\draw[->] (0,-1.5) -- (0,1.5) coordinate (y axis);
\draw (0,0) circle [radius=1cm];
\draw [red,very thick] (30:1cm) -- node[left=1pt,fill=white] {$\sin \alpha$}
    +(0,-0.5);
\draw [blue,very thick] (30:1cm) ++(0,-0.5) -- node[below=2pt,fill=white] {$
    \cos \alpha$} (0,0);
\draw [orange,very thick] (1,0) -- (1,{tan(30)}) node[right=1pt,fill=white]
{$\displaystyle \tan \alpha \color{black}=
\frac{{\color{red}\sin \alpha }}{\color{blue}\cos \alpha}$};
\draw (0,0) -- (1,{tan(30)});
\foreach \x/\xtext in {-1,-0.5/-\frac{1}{2}, 1}
\draw [very thick] (\x cm, -1pt) -- (\x cm, 1pt) node[anchor=north,fill=white]
    {$\xtext$};
\foreach \y in {-1,-0.5,0.5,1}
\draw [very thick] (-1pt,\y cm) -- (1pt,\y cm) node[anchor=east,fill=white] {$
    \y$};
\end{tikzpicture}
```

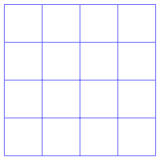


1.8 添加样式 (Add Styles)

Styles 是可用于组织图形绘制方式的预定义选项集。要全局定义样式，可以在文档开头使用\tikzset 命令，用法：

```
\tikzset{style_name./style={options}}
```

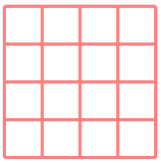
```
\tikzset{blue_thin_lines/.style={color=blue!50,very thin}}
\begin{tikzpicture}
\draw[step=0.5cm, blue_thin_lines] (0,0) grid (2,2);
\end{tikzpicture}
```



为了在局部定义一个样式，我们使用一对方括号 “[ ]” 来定义图片开头的样式，用法：

`[style_name/.style={options}]`

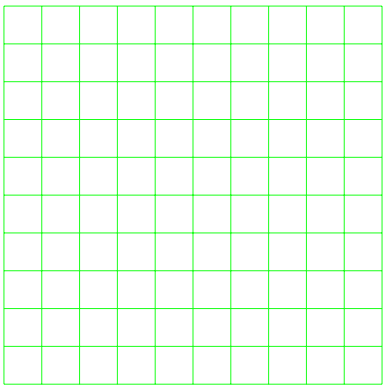
```
\begin{tikzpicture}
[red_thick_lines/.style={color=red!50,very thick}];
\draw[step=0.5cm, red_thick_lines] (0,0) grid (2,2);
\end{tikzpicture}
```



也可以分层定义样式，用法：

`\tikzset{style_name1/.style={style_name2, options}}`

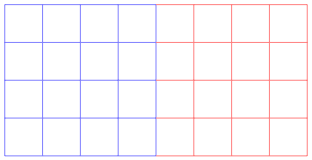
```
\tikzset{green_help_lines/.style={help lines, color=green!90}}
\begin{tikzpicture}
\draw[step=0.5cm, green_help_lines] (0,0) grid (5,5);
\end{tikzpicture}
```



样式也可以与参数一起使用，用法：

```
[style_name/.style={options},style_name/.default={options}]
```

```
\begin{tikzpicture}
[para_color/.style={help lines,color=#1!50}, para_color/.default=blue]
\draw[step=0.5cm, para_color] (0,0) grid (2,2);
\draw[step=0.5cm, para_color=red] (2,0) grid (4,2);
\end{tikzpicture}
```



预定义一个属性集合，到时候直接赋给相应的实体，TikZ 本身就是个宏，因此它为我们提供了强大的属性定义功能，来看这段代码：

```
\begin{tikzpicture}
[LNode/.style={circle, draw=blue!50, fill=blue!20, very thick, minimum size=10
    mm}
\node[LNode] (n1) at (0, 0){$\int x \mathrm{d}x$};
\end{tikzpicture}
```

$$\int_a^b f(x)\mathrm{d}x$$

1.9 Draw options and context

1.9.1 绘图选项

有一些绘图选项可以用来控制颜色、厚度和线条类型。

- 颜色：blue, black, brown, cyan, gray, green, lightgray, lime, magenta, orange, pink, purple, red, yellow, teal, violet, white.

  注意：颜色是可以混合，比如这种颜色命令 `[blue!40!white]` 意味着 40% 蓝色和 60% 白色混合在一起。
- 厚度：ultra thin —, very thin —, thin —, semithick —, thick —, very thick —, ultra thick —.

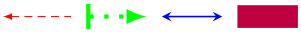
  注意：`[help lines]=[gray,very thin]`. 线厚度也可以通过`[line width]` 选择，例如 `[line width=0.5cm]`.
- 线型：loosely dashed - -, dashed - -, densely dashed - -, loosely dotted · ·, dotted ·····, densely dotted ·····.
- 箭头：`<-` ←, `<<-` ↞, `<-|` ←|, `<<-|` ↞|, `->` →, `->>` ↠, `|->` ↦, `|->>` ↦↠, `<->` ↔, `<<->>` ↔.

  注意: 你也可以添加`>=stealth` 到选项中，它可将箭头更改为'stealth-like' 的样式。

用法如下：

```
\draw[color, thickness, line type, arrow] (x1,y1) -- (x2,y2);
```

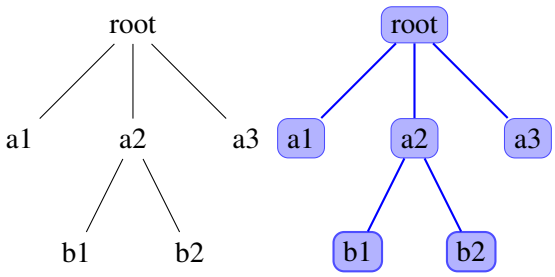
```
\begin{tikzpicture}
\draw[red, very thin, densely dashed, <-] (0,0) -- (0.9,0);
\draw[green, ultra thick, loosely dotted, |->] (1.1,0) -- (1.9,0);
\draw[blue, semithick, <->, >=stealth] (2.1,0) -- (2.9,0);
\draw[purple, line width=0.3cm] (3.1,0) -- (3.9,0);
\end{tikzpicture}
```



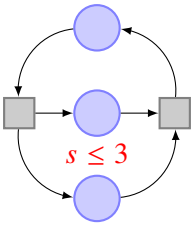
1.9.2 node 树

node 结点不但可以用于添加标识，还可以来绘制树形图，下面看一个例子，两个可作个对比，后面对前加了个样式：

```
\begin{tikzpicture}
  \node {root}
  child {node {a1}}
  child {node {a2}
  child {node {b1}}
  child {node {b2}}}
  child {node {a3}};
\end{tikzpicture}
\begin{tikzpicture}
[every node/.style={fill=blue!30,draw=blue!70,rounded corners},
edge from parent/.style={blue,thick,draw}]
  \node {root}
  child {node {a1}}
  child {node {a2}
  child {node {b1}}
  child {node {b2}}}
  child {node {a3}};
\end{tikzpicture}
```



```
\tikzset{place/.style={circle,draw=blue!50,fill=blue!20,
thick,inner sep=0pt,minimum size=6mm}}
\tikzset{transition/.style={rectangle,draw=black!50,
fill=black!20,thick,inner sep=0pt,minimum size=4mm}}
\tikzset{every label/.style=red}
\begin{tikzpicture}[bend angle=45]
\node[place] (waiting) {};
\node[place] (critical) [below=of waiting] {};
\node[place](semaphore) [below=of critical,label=above:$s\le3$] {};
\node[transition](leave critical) [right=of critical]{};
\node[transition] (enter critical)[left=of critical]{};
\draw [->] (enter critical) to (critical);
\draw [->] (waiting) to [bend right] (enter critical);
\draw [->] (enter critical) to [bend right] (semaphore);
\draw [->] (semaphore) to [bend right] (leave critical);
\draw [->] (critical) to (leave critical);
\draw [->] (leave critical) to [bend right] (waiting);
\end{tikzpicture}
```



1.9.3 scope 环境

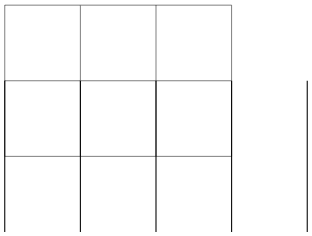
scope 环境就是作用域控制，一个局域环境，参数只影响内部，外部的参数也影响不进来，不过值得一提的是，定义的点外面也可以用。scope 环境一个有用的特性的里面的 clip 命令不会影响到外面。其用法：

```
\begin{scope}[options]
% only apply graphic options inside this scope, but not to anything outside.
\end{scope}
```

```
\begin{tikzpicture}[ultra thick]
\draw (0,0) -- (0,1);
\begin{scope}[thin]
\draw (1,0) -- (1,1);
\end{scope}
\draw (2,0) -- (2,1);
\end{tikzpicture}
```

1.9.4 迭代语句

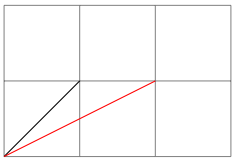
```
\begin{tikzpicture}
\draw[help lines] (0,0) grid (3,2);
\foreach \x in {0,1,...,4}
\draw[xshift=\x cm] (0,-1) -- (0,1);
\end{tikzpicture}
```



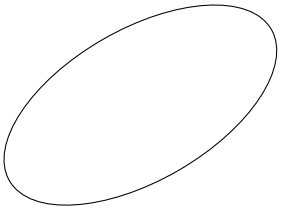
其中... 表示一直这样有规律下去生成迭代列表。迭代语句有很多用法，详见后面的具体例子。

1.9.5 其它

1. 平移：xshift ，x 坐标轴平移。yshift ，y 坐标轴平移。rotate ，旋转。注意 xshift 默认的单位并不是 cm，如果要单位是 cm 需要写出来；



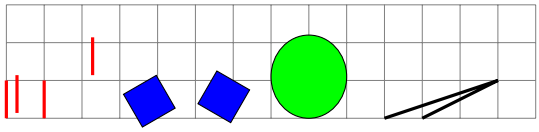
2. 旋转：后面加上可选项 rotate=30 即可，意思是图形逆时针旋转 30 度；



3. 反对称：xscale=-1 或者 yscale=-1 就刚好相对 y 轴或 x 轴反对称；
4. 翻转：例子如下

```
\begin{tikzpicture}
\draw[help lines, step=0.5] (0,0) grid (7,1.5);
\draw[red, very thick] (0,0) -- (0,0.5)
[shift={(4pt,2pt)}] (0,0) -- (0,0.5);
```

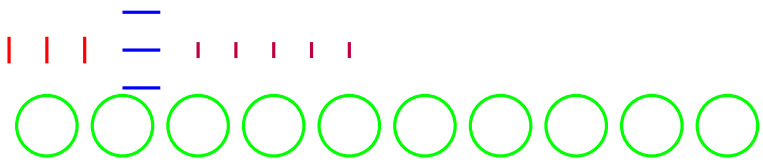
```
\draw[red, very thick] (0.5,0) -- (0.5,0.5)
[shift={+(4pt,2pt)}] (0.5,0) -- (0.5,0.5);
\draw[rotate=30,fill=blue] (1.5,-1) rectangle (2,-0.5);
\draw[rotate around={60:(3,0.5)},fill=blue] (2.5,0.25) rectangle
    (3,0.75);
\draw[xscale=1,yscale=1.1,fill=green] (4,0.5) circle (0.5);
\draw[xslant=2,very thick] (5,0) -- (5.5,0.5) -- (5.5,0);
\end{tikzpicture}
```



5. 循环并行：其用法

```
\foreach \variable in {list of values}{
\commands ;
```

```
\begin{tikzpicture}
\foreach \x in {-0.5cm,0cm,0.5cm}{
\draw[red,very thick] (\x,-5pt) -- (\x,5pt);
}
\foreach \y in {-0.5cm,0cm,0.5cm}{
\draw[blue,very thick] (1cm,\y) -- (1.5cm,\y);
}
\foreach \x in {0,...,9}{
\draw[green,very thick] (\x,-1) circle (0.4cm);
}
\foreach \x in {2,2.5,...,4}{
\draw[purple,very thick] (\x cm,-3pt) -- (\x cm,3pt);
}
\end{tikzpicture}
```



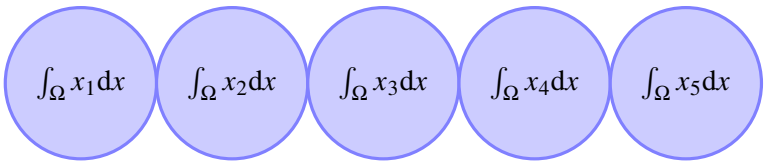
```
 \begin{tikzpicture}
\foreach \x in {1,2,...,5,7,8,...,12}{
\foreach \y in {1,...,5}{
\draw (\x,\y) +(-0.5,-0.5) rectangle +(0.5,0.5);
\draw (\x,\y) node{\x,\y};
}
```

```
}
\end{tikzpicture}
```

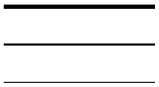
1,5	2,5	3,5	4,5	5,5
1,4	2,4	3,4	4,4	5,4
1,3	2,3	3,3	4,3	5,3
1,2	2,2	3,2	4,2	5,2
1,1	2,1	3,1	4,1	5,1

7,5	8,5	9,5	10,5	11,5	12,5
7,4	8,4	9,4	10,4	11,4	12,4
7,3	8,3	9,3	10,3	11,3	12,3
7,2	8,2	9,2	10,2	11,2	12,2
7,1	8,1	9,1	10,1	11,1	12,1

```
\begin{tikzpicture}
[L1Node/.style={circle, draw=blue!50, fill=blue!20, very thick, minimum
    size=10mm},
L2Node/.style={rectangle,draw=green!50,fill=green!20,very thick, minimum
     size=10mm}]
\foreach \x in {1,...,5}
\node[L1Node] (w1_\x) at (2*\x, 0){$\int_\Omega x_\x$};
\end{tikzpicture}
```



6. 确定路径：
   - 线条：path 路径是最基本的命令，draw 命令等价于 `\path[draw]`，fill 命令等价于 `\path[fill]`，filldraw 命令等价于 `\path[draw,fill]`，其他 clip，shade 命令情况类似。
   - 虚线和点线：线条除了之前说的 dashed 和 dotted 两种样式之外，还有 loosely dashed，densely dashed 和 loosely dotted，densely dotted;
   - 线条的粗细：其他选项还有 ultra thin , very thin, thin, semithick, very thick, ultra thick

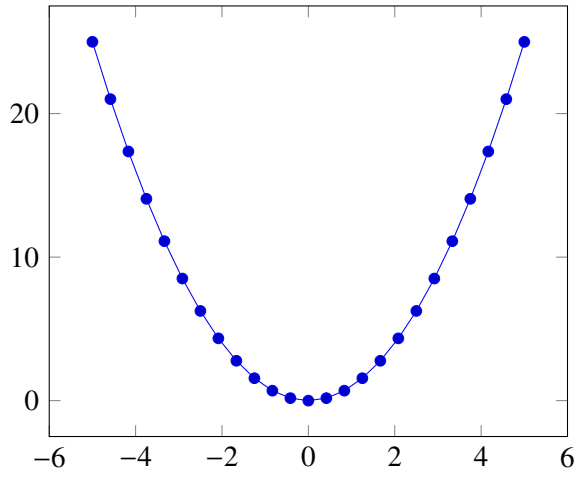


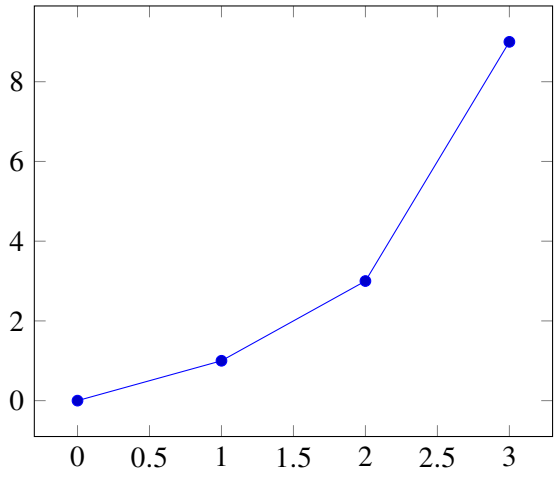
或者直接通过可选项 line width 来定义：

1.10 pgfplots 宏包

pgfplots 宏包真是太好了，有时甚至画一个基本的坐标轴都懒得动用其他宏包命令了，我可以直接调用一个 axis 环境和进行一些简单的优化即可。当然就作为坐标轴作图可能总是用 pgfplots 宏包可能会稍显单调，但如果要求不是特别高的确实用 pgfplots 宏包会基于坐标轴的各个图形非常的称心如意，比如下面两个例子直接画函数与根据数据点来绘制：

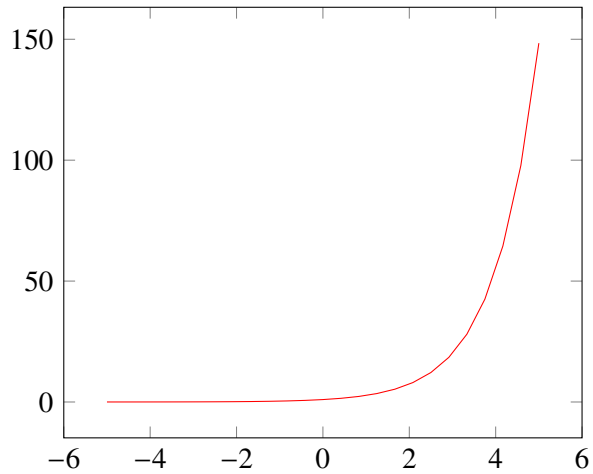
```
\begin{tikzpicture}
\begin{axis}
\addplot {x^2};
\end{axis}
\end{tikzpicture}
\begin{tikzpicture}
\begin{axis}
\addplot coordinates
{(0,0)
(1,1)
(2,3)
(3,9)};
\end{axis}
\end{tikzpicture}
```

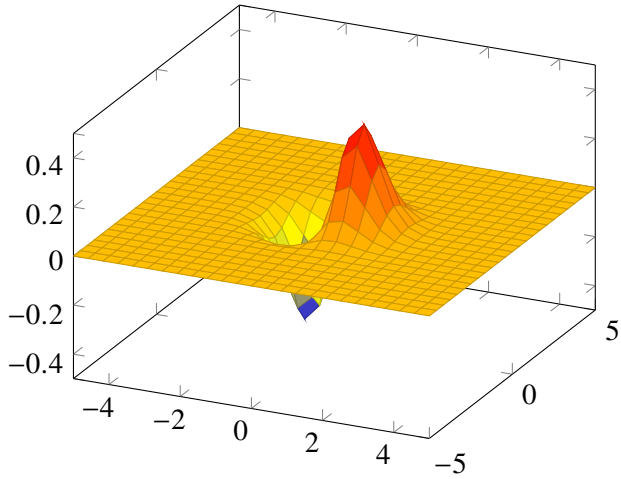





```
\begin{tikzpicture}
\begin{axis}
\addplot[color=red]{exp(x)};
\end{axis}
\end{tikzpicture}
%Here ends the furst plot
\hskip 5pt
%Here begins the 3d plot
\begin{tikzpicture}
```

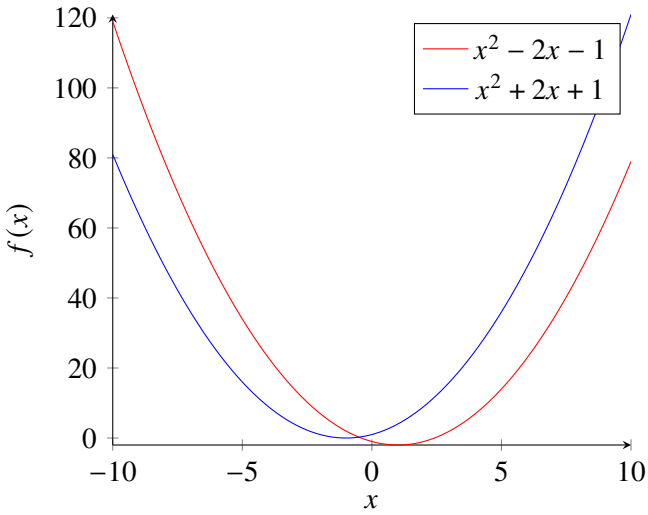
```
\begin{axis}
\addplot3[
surf,
]
{exp(-x^2-y^2)*x};
\end{axis}
\end{tikzpicture}
```



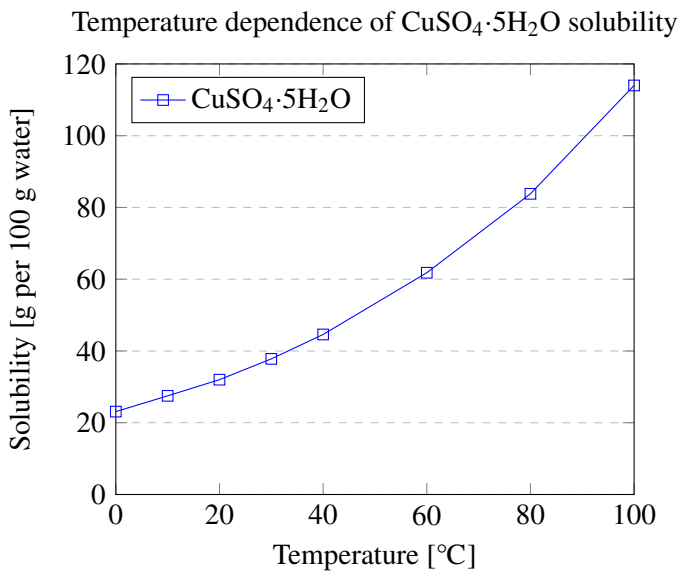



```
\begin{tikzpicture}
\begin{axis}[
axis lines = left,
xlabel = $x$,
ylabel = {$f(x)$},
]
%Below the red parabola is defined
\addplot [
domain=-10:10,
samples=100,
color=red,
]
{x^2 - 2*x - 1};
\addlegendentry{$x^2 - 2x - 1$}
%Here the blue parabloa is defined
\addplot [
domain=-10:10,
samples=100,
color=blue,
]
{x^2 + 2*x + 1};
\addlegendentry{$x^2 + 2x + 1$}

\end{axis}
```

```
\end{tikzpicture}
```

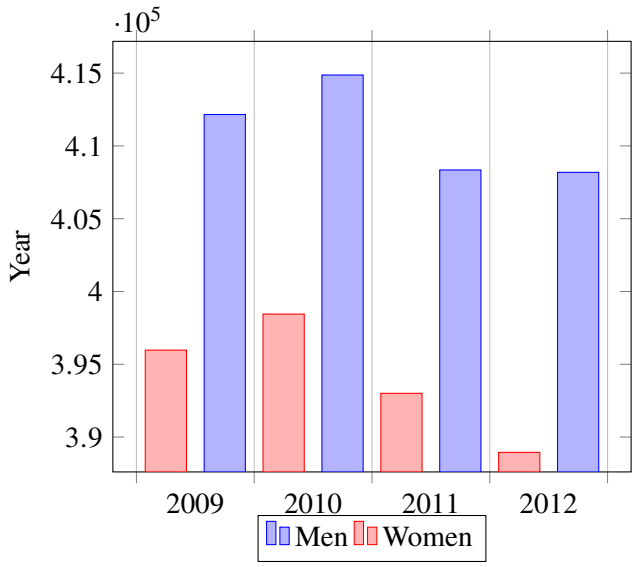



```
\begin{tikzpicture}
\begin{axis}[
title={Temperature dependence of CuSO$_4\cdot$5H$_2$O solubility},
xlabel={Temperature [\textcelsius]},
ylabel={Solubility [g per 100 g water]},
xmin=0, xmax=100,
ymin=0, ymax=120,
xtick={0,20,40,60,80,100},
ytick={0,20,40,60,80,100,120},
legend pos=north west,
ymajorgrids=true,
grid style=dashed,
]

\addplot[
color=blue,
mark=square,
]
coordinates {
(0,23.1)(10,27.5)(20,32)(30,37.8)(40,44.6)(60,61.8)(80,83.8)(100,114)
};
\legend{CuSO$_4\cdot$5H$_2$O}

\end{axis}
\end{tikzpicture}
```

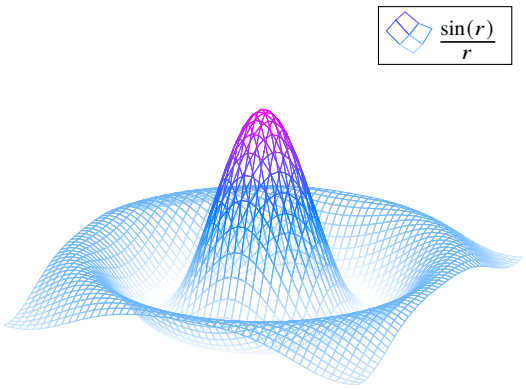



```
\begin{tikzpicture}
\begin{axis}[
x tick label style={
/pgf/number format/1000 sep=},
ylabel=Year,
enlargelimits=0.05,
legend style={at={(0.5,-0.1)},
anchor=north,legend columns=-1},
ybar interval=0.7,
]
\addplot
coordinates {(2012,408184) (2011,408348)
(2010,414870) (2009,412156) (2008,415 838)};
\addplot
coordinates {(2012,388950) (2011,393007)
(2010,398449) (2009,395972) (2008,398866)};
\legend{Men,Women}
\end{axis}
\end{tikzpicture}
\begin{tikzpicture}
\begin{axis}[
title=Exmple using the mesh parameter,
hide axis,
colormap/cool,
]
\addplot3[
mesh,
samples=50,
domain=-8:8,
```

```
]
{sin(deg(sqrt(x^2+y^2)))/sqrt(x^2+y^2)};
\addlegendentry{$\frac{\sin(r)}{r}$}
\end{axis}
\end{tikzpicture}
```

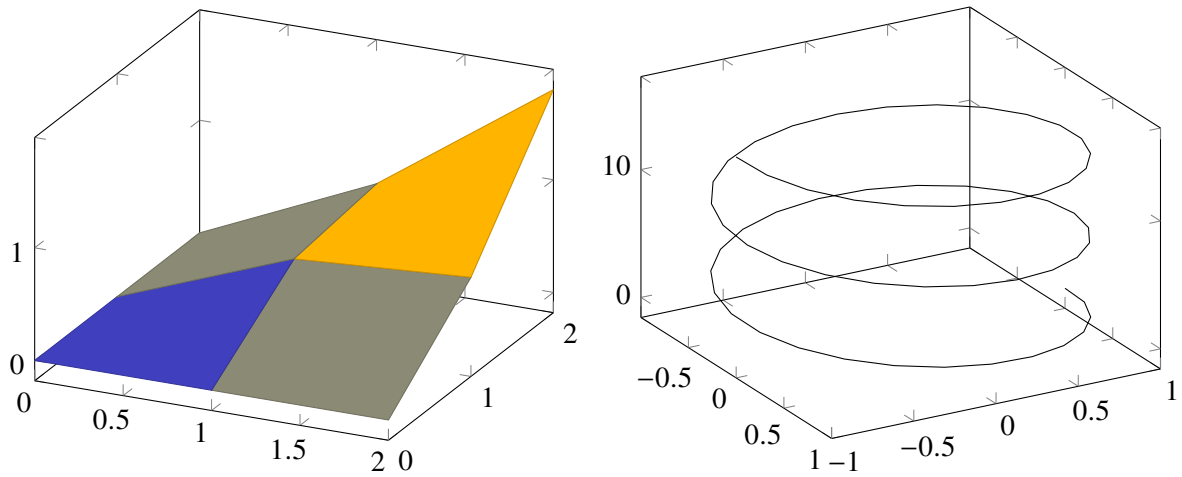


Exmple using the mesh parameter




```
\begin{tikzpicture}
\begin{axis}
\addplot3[
surf,
]
coordinates {
(0,0,0) (0,1,0) (0,2,0)

(1,0,0) (1,1,0.6) (1,2,0.7)

(2,0,0) (2,1,0.7) (2,2,1.8)
};
\end{axis}
\end{tikzpicture}
\begin{tikzpicture}
\begin{axis}
[
view={60}{30},
]
\addplot3[
domain=0:5*pi,
samples = 60,
samples y=0,
]
```

```
({sin(deg(x))},
{cos(deg(x))},
{x});
\end{axis}
\end{tikzpicture}
```



1.11 电路图

电路基本符号，具体详细看CircuiTikZ

1. battery : 电池
2. bulb : 灯泡 ⊗
3. make contact : 开关
4. make contact : 开关另一种形式额外选项 [set make contact graphic= var make contact IEC graphic]
5. resistor 电阻 (加上选项 [ohm=20k] 则上面写上电阻数值)
6. contact 电线交点 •
7. current direction to 路径上加上电流方向 (如果是 [**current direction'**] 则方向反向。)

连线问题：各个元器件之间的连线除了一般的 – 连直线外，还可以通过 -| 或者 |- 来处理垂直拐线的问题，其中 -| 你可以理解为从第一个点先横着走再竖着走，而 |- 你可以理解为先从第一个点竖着走再横着走。

翻转问题：四个基本的选项 [point up ,point down, point left, point right]，分别是朝上，朝下，朝左和朝右，其他复杂的角度的处理方法不是用 rotate 选项，而是在路径上加上上面的电路符号选项，这样那些元器件会自动跟随路径对齐的。

1.12 小总结

TikZ 只是一个前端 (frontend)，画图功能通过调用底层 PGF 宏包完成。

1. 简易的 tikz 环境：一个分号表示画图的结束;

```
\tikz ...;
```

可以有多个画图语句

```
\tikz{...;...;...;}
```

2. 整体缩放图形倍数：magniication=1 为原始大小

```
\begin{tikzpicture}[scale=<magnification>]
......
\end{tikzpicture}
```

3. 自定义一个图形，可方便重复使用：`\def\<name>{<a path>}`

```
%自定义一个正方形
\def\rectangle{-- ++(1,0) -- ++(0,1) -- ++(-1,0) -- cycle}
```

4. 自定义一组样式，方便重复使用，并可设置参数：`<name>/.style={<attributes>}`

```
help line/.style={very thin, color=#1red!20!blue!20,rounded corners=2pt}
```

5. 最基本画图命令：`\path`

```
\draw=\path[draw], \draw[color=<color>]=\path[draw=<color>]
\fill=\path[fill], \fill[color=<color>]=\path[fill=<color>]
\filldraw=\path[fill, draw]
\clip=\path[clip]
```

6. 定位：

```
+(x,y) 在之前的画笔点的基础上偏移(x,y)，但并不改变画笔点;
++(x,y)同样是偏移(x,y)，但把偏移之后的点做为新的画笔点。
```

7. scope 环境：需要注意`\clip`的作用范围是从其语句之后一直到当前所在 scope 的结束

```
\begin{scope}[<sequence of atttibutes>]
......
\end{scope}
```

8. 几个典例：
   - 球坐标 (angle:radius)
   - 弧 (point at the initial angle) arc(inital angle:terminal angle:radius)
   - 圆 (center) circle (radius)
   - 椭圆 (center) ellipse (x radius and y radius)
   - 矩形 (point 1) rectangle (point 2)
   - 交点 (intersection of <line1> and <line2>)

9. 可用选项：
   - 箭头：->, -», <-, «-, <->
   - 旋转：rotate=<angle>
   - 圆角：rounded corners=<x pt>
   - 颜色：color=<color1!percentage1><!color2!percentage2>...
   - 虚线：dashed, loosely dashed, densely dashed, dotted, loosely dotted, densely dotted
   - 宽度：very thin, thin (正常宽度), thick, very thick
10. 缩放：

```
[scale=<magnification>],[xscale=<magnification>],[yscale=<manification
    >].
```

其中若 <magnification> 取值为实数，其绝对值表示缩放的倍数；若 <magnification> 是负数，表示进行翻转；若 xscale 是负数则左右翻转，yscale 是负数则上下翻转，scale 是负数则同时翻转。

11. 偏移：

```
[shift=<point>], [shift=+<point>],[xshift=<d_x>], [yshift=<d_y>].
```

将后面的图加上一个矢量。其中 [shift=+<point>] 表示，此加上的矢量为前一个画笔点 +<point>。

参考文献

[1] http://hohei3108.hatenablog.com/entry/2017/10/15/000611

[2] https://blog.csdn.net/stereohomology/article/details/24266409

[3] LaTeX 科技排版工作室

[4] https://github.com/Chris7462/TikZ

[5] http://blog.sina.com.cn/s/blog_72277faf0100xqz6.html