tkz-base

I’d like to thank Till Tantau for creating the wonderful tool TikZ.

I thank Yves Combe for sharing his work on the protractor and the compass constructions. I would also like to thank, David Arnold who corrected a lot of errors and tested many examples, Wolfgang Büchel who also corrected errors and built great scripts to get the example files, John Kitzmiller and Dimitri Kapetas for their examples, and finally Gaétan Marris for his remarks and corrections.

You will find many examples on my site: altermundus.fr.

You can send your remarks, and reports on errors you find, to the following address: Alain Matthes. This file can be redistributed and/or modified under the terms of the \TeX\ Project Public License Distributed from CTAN archives.
Contents

1 News and presentation ........................................... 7
2 Installation .......................................................... 8
  2.1 Files present .................................................... 8
3 Compilation of examples ........................................... 9
  3.1 Installation test ................................................ 9
  3.2 \texttt{xfp} and \texttt{numprint} .................................. 9
4 Presentation of \texttt{tkz-base} ................................... 10
  4.1 Example that poses a problem ................................ 10
  4.2 The role of \texttt{tkz-base} .................................... 10
  4.3 Syntax of \texttt{tkz-base} ....................................... 11
5 Initialization \texttt{\tkzInit} ...................................... 11
  5.1 The main macro \texttt{\tkzInit} ................................ 11
    5.1.1 Changing the drawing size with \texttt{\tkzInit} ....... 12
    5.1.2 Role of \texttt{xstep}, \texttt{ystep} ......................... 12
  5.2 Another example with \texttt{xstep} and \texttt{ystep} .......... 13
    5.2.1 Customized origin. ....................................... 13
    5.2.2 Use of decimals ........................................... 13
    5.2.3 Negative values .......................................... 14
6 Macros for the axes .............................................. 15
  6.1 \texttt{\tkzDrawX} ................................................ 15
    6.1.1 No tick, no label ......................................... 15
    6.1.2 Label placement .......................................... 15
    6.1.3 Label and Axis Colour ................................... 16
    6.1.4 Option \texttt{right space} ................................ 16
    6.1.5 Trigonometric axis with the option \texttt{trig=1} ....... 16
    6.1.6 Trigonometric axis with the option \texttt{trig=2} ....... 16
  6.2 \texttt{\tkzLabelX} ................................................ 16
    6.2.1 Position of the graduations ............................. 17
    6.2.2 Position of the graduations with \texttt{xlabel style} .. 17
    6.2.3 Dates with \texttt{np off} .................................. 17
    6.2.4 \texttt{frac} .................................................. 17
    6.2.5 \texttt{trig} .................................................. 17
    6.2.6 Graduations size ......................................... 17
    6.2.7 Colour of the graduations ............................... 18
    6.2.8 Axis drawings before the graduation .................... 18
    6.2.9 Graduations (except originally) prior to tracings .... 19
    6.2.10 Only positive graduations before drawings .......... 19
    6.2.11 No graduations at the origin ............................ 19
  6.3 \texttt{\tkzAxeX} .................................................. 19
    6.3.1 Example with \texttt{\tkzAxeX} ............................. 20
    6.3.2 Use of \texttt{pi} and \texttt{\tkzAxeX} ....................... 20
    6.3.3 Option \texttt{frac} and \texttt{trig} ......................... 20
  6.4 \texttt{\tkzDrawY} .................................................. 20
  6.5 \texttt{\tkzLabelY} .................................................. 21
  6.6 \texttt{\tkzAxeY} .................................................. 21
  6.7 \texttt{\tkzAxeXY} .................................................. 21
    6.7.1 Colour of axes, graduations ............................ 21
    6.7.2 Option \texttt{label={}} ................................... 22
    6.7.3 Option \texttt{swap} ....................................... 22
6.8 \tkzDrawXY ................................................................. 22
   6.8.1 Common colour and empty labels ................................ 22
   6.8.2 Two trigonometric axes ............................................ 23
6.9 \tkzLabelXY .............................................................. 23
6.10 Changing values by axis default ................................... 23
   6.10.1 Changing the default axes ....................................... 24
7 Use of \tkzGrid .............................................................. 25
   7.0.1 \tkzGrid and the option sub ...................................... 25
   7.0.2 Option sub .................................................................. 25
   7.0.3 Almost Default .......................................................... 26
   7.0.4 Under the grid, too, option sub .................................... 26
   7.0.5 Grid change ............................................................. 26
   7.0.6 Option xstep, xstep, subxstep and subystep .................. 26
   7.0.7 With large intervals .................................................. 27
   7.0.8 \tkzGrid and the arguments .......................................... 27
   7.0.9 Use of pi with \tkzGrid .............................................. 27
   7.0.10 Changing the default axes ......................................... 28
   7.0.11 Use of a repetition grid ............................................ 28
8 The points ....................................................................... 29
   8.1 Defining a point in Cartesian coordinates: \tkzDefPoint .......... 29
      8.1.1 Use of shift .......................................................... 29
   8.2 Placing a label with the library quotes ............................ 29
      8.2.1 Rotation with shift and scope .................................... 30
      8.2.2 Forms and coordinates ............................................. 30
      8.2.3 Scope et \tkzDefPoint .............................................. 30
   8.3 Definition of points in Cartesian coordinates: \tkzDefPoints ... 31
      8.3.1 Definition of points ................................................ 31
   8.4 Point relative to another: \tkzDefShiftPoint .................... 31
      8.4.1 Example with \tkzDefShiftPoint .................................. 31
   8.5 Point relative to another: \tkzDefShiftPointCoord ............. 32
      8.5.1 Equilateral triangle with \tkzDefShiftPointCoord .......... 32
      8.5.2 Isosceles triangle with \tkzDefShiftPointCoord ............ 32
   8.6 Drawing a point \tkzDrawPoint ....................................... 33
      8.6.1 Default stitch style ................................................ 33
      8.6.2 Changing the style .................................................. 33
      8.6.3 Example of point plots ........................................... 34
   8.7 Drawing points \tkzDrawPoints ....................................... 34
      8.7.1 Example with \tkzDefPoint and \tkzDrawPoints ............ 34
      8.7.2 More complex example ............................................. 35
   8.8 Add a label to a point \tkzLabelPoint ................................ 35
      8.8.1 Example with \tkzLabelPoint ..................................... 35
      8.8.2 Label and reference .............................................. 35
   8.9 Add labels to points \tkzLabelPoints ............................... 36
      8.9.1 Example with \tkzLabelPoints ................................... 36
   8.10 Automatic position of labels \tkzAutoLabelPoints ............. 36
      8.10.1 Example with \tkzAutoLabelPoints ............................ 36
      8.10.2 Example with \tkzAutoLabelPoints ............................ 37
   8.11 Point style with \tkzSetUpPoint .................................... 37
      8.11.1 Simple example with \tkzSetUpPoint ......................... 38
      8.11.2 Example with \tkzSetUpPoint ................................... 38
      8.11.3 Using \tkzSetUpPoint in a group .............................. 38
   8.12 Show point coordinates ............................................. 38
      8.12.1 Default styles ..................................................... 39
      8.12.2 Example with \tkzPointShowCoord ............................. 39
<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.12.3 Example with \tkzPointShowCoord and xstep</td>
</tr>
<tr>
<td>8.13 \tkzDefSetOfPoints</td>
</tr>
<tr>
<td>8.13.1 Creating a scatter plot with \tkzDefSetOfPoints</td>
</tr>
<tr>
<td>9 Style Use</td>
</tr>
<tr>
<td>9.1 Modification of \tkz-base</td>
</tr>
<tr>
<td>9.2 Use \tikzset</td>
</tr>
<tr>
<td>9.3 Configuration Macros</td>
</tr>
<tr>
<td>10 Bounding box management</td>
</tr>
<tr>
<td>10.1 \tkzShowBB</td>
</tr>
<tr>
<td>10.1.1 Example with \tkzShowBB</td>
</tr>
<tr>
<td>10.2 \tkzClipBB</td>
</tr>
<tr>
<td>10.2.1 Example with \tkzClipBB and the bisectors</td>
</tr>
<tr>
<td>10.3 \tkzSetBB</td>
</tr>
<tr>
<td>10.3.1 Example with \tkzShowBB</td>
</tr>
<tr>
<td>10.4 \tkzSaveBB</td>
</tr>
<tr>
<td>10.5 \tkzRestoreBB</td>
</tr>
<tr>
<td>10.5.1 Example of the use of \tkzRestoreBB</td>
</tr>
<tr>
<td>10.6 \tkzClip</td>
</tr>
<tr>
<td>10.6.1 First example with \tkzClip</td>
</tr>
<tr>
<td>10.6.2 Second example with \tkzClip</td>
</tr>
<tr>
<td>10.6.3 \tkzClip et l’option space</td>
</tr>
<tr>
<td>10.7 Reverse clip: \tkzreverseclip</td>
</tr>
<tr>
<td>10.7.1 Example with \tkzClipOutPolygon</td>
</tr>
<tr>
<td>10.8 Options from Ti\textsc{k}Z: trim left or right</td>
</tr>
<tr>
<td>10.9 Ti\textsc{k}Z.Controls \pgfinterruptboundingbox and \endpgfinterruptboundingbox</td>
</tr>
<tr>
<td>11 Use Additional Objects or Tools</td>
</tr>
<tr>
<td>12 Using an orthogonal coordinate system</td>
</tr>
<tr>
<td>12.1 Repère avec \tkzRep</td>
</tr>
<tr>
<td>12.1.1 Some modifiable styles</td>
</tr>
<tr>
<td>12.1.2 Example of use</td>
</tr>
<tr>
<td>13 Lines parallel to the axes</td>
</tr>
<tr>
<td>13.1 Draw a horizontal line with \tkzHLine</td>
</tr>
<tr>
<td>13.1.1 Horizontal line</td>
</tr>
<tr>
<td>13.1.2 Horizontal line and value calculated by xfp</td>
</tr>
<tr>
<td>13.2 Horizontal lines with \tkzHLines</td>
</tr>
<tr>
<td>13.2.1 Horizontal lines</td>
</tr>
<tr>
<td>13.3 Draw a vertical line with \tkzVLine</td>
</tr>
<tr>
<td>13.3.1 Vertical line</td>
</tr>
<tr>
<td>13.3.2 Vertical line and value calculated by xfp</td>
</tr>
<tr>
<td>13.4 Vertical lines with \tkzVLines</td>
</tr>
<tr>
<td>13.4.1 Vertical lines</td>
</tr>
<tr>
<td>14 Ticks on the axes</td>
</tr>
<tr>
<td>14.1 Drawing one tick on the abscissa axis \tkzHTick</td>
</tr>
<tr>
<td>14.1.1 Example</td>
</tr>
<tr>
<td>14.2 Drawing ticks on the ordinate axis \tkzHTicks</td>
</tr>
<tr>
<td>14.3 Drawing one tick on the abscissa axis \tkzVTick</td>
</tr>
<tr>
<td>14.4 Drawing ticks on the abscissa axis \tkzVTicks</td>
</tr>
<tr>
<td>15 Marks or symbols</td>
</tr>
<tr>
<td>15.1 \tkzDrawSetOfPoints</td>
</tr>
<tr>
<td>15.1.1 Drawing of a scatter plot with \tkzDrawSetOfPoints</td>
</tr>
</tbody>
</table>
15.2 \texttt{\tkzJoinSetOfPoints} ................................................................. 55
15.2.1 Link the points of a scatter plot with \texttt{\tkzJoinSetOfPoints} .............. 56
15.2.2 Using the points of a scatter plot ....................................................... 56
15.3 \texttt{\tkzSetUpMark} .......................................................... 56
15.3.1 Two scatter plots ............................................................................. 57
15.4 \texttt{\tkzDrawMark} .......................................................... 57
15.4.1 Ball; use of \texttt{\tkzDrawMarks} ......................................................... 57
15.5 \texttt{\tkzDrawMarks} .......................................................... 57
15.5.1 Mark and plot; use of \texttt{\tkzDrawMarks} .......................................... 58

16 Texts and Legends ................................................................................. 59
16.1 Placing a title ...................................................................................... 59
16.1.1 A title .............................................................................................. 59
16.1.2 Draft ............................................................................................... 59
16.1.3 Text with a period ............................................................................ 59
16.1.4 Text format ...................................................................................... 60
16.2 Placing legends .................................................................................. 60
16.2.1 Legends with symbols ..................................................................... 61

17 FAQ ........................................................................................................ 62
17.1 General Questions ............................................................................... 62
17.2 Most common errors ........................................................................... 62

Index ......................................................................................................... 63
1 News and presentation

This package is the foundation of the \texttt{tkz-euclide} and \texttt{tkz-fct} in particular. It provides a Cartesian system that will be defined by the macro \texttt{\tkzInit}. The package has been modified and object transfers between \texttt{tkz-base} and \texttt{tkz-euclide} have been performed. In the future, the definition macros will be isolated.

The main novelty is the recent replacement of the \texttt{fp} package by \texttt{xfp}. The appearance of this one is a step towards version 3 of \LaTeX. The next step will be the creation of a new package.

Here are some of the changes. The \texttt{tkz-euclide} package brings more new features.

\begin{itemize}
\item Code Improvement.
\item Bug correction.
\item The bounding box is now controlled in each macro (hopefully) to avoid the use of \texttt{\tkzInit} followed by \texttt{\tkzClip}.
\item Addition of macros for the "bounding box": \texttt{\tkzSaveBB \tkzClipBB}.
\item Logically most macros accept TikZ options. So I removed the "duplicate" options.
\item Removing the option "label options".
\item Random points are now in \texttt{tkz-euclide} and the macro \texttt{\tkzGetRandPointOn} is replaced by \texttt{\tkzDefRandPointOn}. For homogeneity reasons, the points must be retrieved with \texttt{\tkzGetPoint}.
\item The options \texttt{end} and \texttt{start} which allowed to give a label to a line are removed. You must now use the macro \texttt{\tkzLabelLine}.
\item Introduction of the libraries \texttt{quotes} and \texttt{angles} it allows to give a label to a point even if I am not in favour of this practice.
\item Appearance of the macro \texttt{\usetkztool} which allows to load new "tools".
\end{itemize}
2 Installation

`tkz-base` is now on the server of the CTAN\(^1\). If you want to test a beta version, just put the following files in a texmf folder that your system will be able to find.

2.1 Files present

Before testing the installation, you can verify that the `tkzbase` folder contains the following files:

- `tkz-base.cfg`
- `tkz-base.sty`
- `tkz-lib-marks.tex`
- `tkz-obj-axes.tex`
- `tkz-obj-grids.tex`
- `tkz-obj-marks.tex`
- `tkz-obj-points.tex`
- `tkz-obj-rep.tex`
- `tkz-tools-arith.tex`
- `tkz-tools-base.tex`
- `tkz-tools-BB.tex`
- `tkz-tools-math.tex`
- `tkz-tools-misc.tex`
- `tkz-tools-modules.tex`
- `tkz-tools-print.tex`
- `tkz-tools-text.tex`
- `tkz-tools-utilities.tex`

The one with the main macros is `tkz-tools-base.tex`, it's called `tkz-base` which handles all the files. The various tools are in files beginning with `tkz-tools`, the mathematical objects created are in files whose name has for prefix `tkz-obj`. Finally `tkz-base.cfg` whose presence is not mandatory allows to modify many default values.

Moreover, Ti\kZ is loaded with the following libraries:

```
\usetikzlibrary{angles, arrows, arrows.meta, babel, calc, decorations, decorations.markings, decorations.pathreplacing, intersections, patterns, plotmarks, positioning, quotes, shapes.misc, shapes.misc, through}
```

\(^1\) `tkz-base` is part of TeXLive and `tlmgr` allows to install it, it is also part of MikTeX under Windows

---

`tkz-base`  
AlterMundus
3 Compilation of examples

3.1 Installation test

The code below allows you to test your installation of \texttt{tkz-base}. Please note that \texttt{xfp} as well as \texttt{numprint} must be present as well as version 3.01 (or higher) of \texttt{pgf}. All examples and this documentation have been compiled using Lua\TeX.

\begin{verbatim}
\documentclass{standalone}
\usepackage{tkz-base}
\begin{document}
\begin{tikzpicture}
\tkzInit[xmax=4,ymax=4]
\tkzGrid
\tkzAxeXY
\end{tikzpicture}
\end{document}
\end{verbatim}

Notes on this test

1. The compilation of this document and examples is obtained with \texttt{luaLaTeX-dev}.
2. In principle, \texttt{tkz-base} is not loaded by the user, it will be loaded by another package like \texttt{tkz-euclide} or \texttt{tkz-fct}. \texttt{tkz-base} loads \texttt{numprint} with the option \texttt{autolanguage}, \texttt{xfp} and of course Ti\texttt{kZ}.
3. Ti\texttt{kZ} seems that version 3 of \texttt{pgf} has fixed those problems. In case of difficulty, it is recommended to load the \texttt{babel} library with \texttt{\usepackage{babel}}. Another possibility is to compile with Lua\TeX.

3.2 \texttt{xfp} and \texttt{numprint}

\texttt{xfp} now replaces \texttt{fp} in this package. One of the advantages for the user is a simplified syntax. It allows to manage calculations on large or very small numbers with precision. This slows down the compilation a bit, so it is better not to overuse it. \texttt{xfp} is used above all, to obtain correct graduations.

\texttt{numprint} was present when I started to write this series of packages, since \texttt{siunitx} has grown and I can understand that some people prefer it. In a future version, I plan to leave the choice of the package for displaying numbers.
4 Presentation of \texttt{tkz-base}

4.1 Example that poses a problem

The following code gives an error

\begin{tikzpicture}
\draw (0,0)--(600,0);
\end{tikzpicture}

\texttt{Latex Error: ... Dimension too large.}

Indeed, the default unit is a centimeter or \LaTeX{} cannot store a dimension greater than 575 cm, which leads to an error. \LaTeX{} however, can store integers up to $2^{31} - 1$, so it is possible to work on integers first and then define the dimensions.

\begin{tikzpicture}[x=0.01 cm]
\draw (0,0)--(600 cm,0);
\end{tikzpicture}

\texttt{Latex Error: ... Dimension too large.}

The previous code still makes an error. Indeed, 600 cm is a dimension and does not take into account the change of unit. Correct is:

\begin{tikzpicture}[x=0.01 cm]
\draw (0,0)--(600,0);
\end{tikzpicture}

This time, the stored dimension is 6 cm which is acceptable. It is possible with \LaTeX{} to handle large whole numbers, but, on the other hand, the dimensions cannot exceed 16,384 pt or approximately 5.75 m.

With \LaTeX{}, it’s also possible to work with the \texttt{xfp} package. This allows him to work at longer intervals, but at the cost of a certain slowness. This is the method I have preferred for some sensitive calculations that require good precision, such as calculations to measure angles or segment length, but it is necessary once a number has been found to assign it to a dimension. We always find the same constraints.

4.2 The role of \texttt{tkz-base}

The following code gives an error not because 6,000,000 is too large, but because 0.000,001 cm is too small.

\texttt{Latex Error:}

\begin{tikzpicture}[x=0.000001 cm]
\coordinate (x) at (6000000,0);
\draw (0,0)--(x);
\end{tikzpicture}

With \texttt{tkz-base}, it will be possible to work with any coordinates, but it will be necessary to use the macros of the package.

\texttt{tkz-base} simplifies the use of different value ranges. This package is used by several of my packages such as \texttt{tkz-tukey}, a package for drawing graphical representations in elementary statistics, \texttt{tkz-fct} which allows to draw graphical representations of functions using \texttt{gnuplot}, as well as with \texttt{tkz-euclide} for Euclidean geometry.

First of all, you should know that it is not necessary to deal with \LaTeX{} with the size of the support (bounding box); however it is sometimes necessary, either to draw a grid, or to draw axes, or to work with a different unit than the centimeter, or finally to control the size of what will be displayed. To do this, you must have prepared the frame in which you are going to work, this is the role of \texttt{tkz-base} and its main macro \texttt{tkzInit}. For example, if you want to work on a 10 cm square, but such that the unit is the dm then you will have to use.
\tkzInit[xmax=1,ymax=1,xstep=0.1,ystep=0.1]

\texttt{xstep=0.1} means that 1 cm represents the 0.1 graduation so the 1 graduation is at 10 cm from the origin. On the other hand, for values of \( x \) between 0 and 10,000 and values of \( y \) between 0 and 100,000, it will be necessary to write

\tkzInit[xmax=10000,ymax=100000,xstep=1000,ystep=10000]

The result is always a 10 cm square.

All this makes little sense for Euclidean geometry, and in this case it is recommended to leave the graphic unit equal to 1 cm. I have not tested whether all macros for Euclidean geometry accept other values than \texttt{xstep=1} and \texttt{ystep=1}. On the other hand, for some drawings, it is interesting to fix the extreme values and to "clip" the definition rectangle in order to control the size of the figure as well as possible.

### 4.3 Syntax of \texttt{tkz-base}

I tried to generalize the following syntax:

- The syntax is close to that of \texttt{\LaTeX}, there's no need for ";"
- all the macros have names beginning with \texttt{tkz};
- braces are used to pass a parameter that will be the reference of an object created by the macro;
- parentheses are used to refer to an object that has already been created or to a coordinate pair;
- square brackets are necessary to pass optional arguments or options, some choices are sometimes mandatory. The use of the comma even in a Math mode requires to be protected in a TeX group;
- blanks (space) are prohibited between \([...]\) and \((...), [...] and {...}), as well as between \((...) and {...}), but it is possible to put spaces between past optional arguments {...}.

### 5 Initialization \texttt{\tkzInit}

#### 5.1 The main macro \texttt{\tkzInit}

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmin</td>
<td>0</td>
<td>minimum value of the abscissae in cm</td>
</tr>
<tr>
<td>xmax</td>
<td>10</td>
<td>maximum value of the abscissae in cm</td>
</tr>
<tr>
<td>xstep</td>
<td>1</td>
<td>difference between two graduations in ( x )</td>
</tr>
<tr>
<td>ymin</td>
<td>0</td>
<td>minimum y-axis value in cm</td>
</tr>
<tr>
<td>ymax</td>
<td>10</td>
<td>maximum y-axis value in cm</td>
</tr>
<tr>
<td>ystep</td>
<td>1</td>
<td>difference between two graduations in ( y )</td>
</tr>
</tbody>
</table>

The role of \texttt{\tkzInit} is to define a orthogonal coordinates system and a rectangular part of the plane in which you will place your drawings using Cartesian coordinates. The coordinates system does not have to be normalized. This macro allows you to define your working environment as with a calculator.
5.1.1 Changing the drawing size with \tkzInit

This macro sets the stage and defines several constants. It is quite possible to make a figure larger than the predefined rectangle. Moreover, as you can see, it is possible to use the commands of TikZ in the middle of those of \tkz but attention to the units! This possibility must be reserved for exceptional cases only.

```latex
\begin{tikzpicture}
\tkzInit[xmax=8,ymax=6]
\tkzGrid
\tkzAxeXY
\draw[blue](-1,0)--(6,7);
\end{tikzpicture}
```

5.1.2 Role of xstep , ystep

\begin{itemize}
\item \textbf{Warning}, a graduation is represented by 1 cm, unless you resize the figure with the \texttt{scale} option. In the example below \texttt{xstep = 2} corresponds to 1 cm, so between 0 and 10, we will need 5 cm. Similarly \texttt{ystep=400}, so between 0 and 800 there are 2 cm. It is not possible to use the options of TikZ, \texttt{x=} and \texttt{y=}.
\end{itemize}

```latex
\begin{tikzpicture}
\tkzInit[xmax=10,xstep=2,ymax=800,ystep=400]
\tkzGrid
\tkzAxeXY
\end{tikzpicture}
```
5.2 Another example with xstep and ystep

\begin{tikzpicture}
\tkzInit[xmax=5,xstep=1,ymax=2,ystep=.5]
\tkzGrid
\tkzAxeXY
\end{tikzpicture}

5.2.1 Customized origin.
It is important to note that you can place a point without calculating anything.

\begin{tikzpicture}
\tkzInit[xmin=20,
xmax=50,
xstep=10,
ymin=5000,
ymax=5150,
ystep=50]
\tkzAxeXY
\tkzDefPoint(30,5100){A}
\tkzDrawPoint(A)
\end{tikzpicture}

5.2.2 Use of decimals
It is preferable to write the different arguments relating to an axis with the same number of decimals. \texttt{numprint} is used to display the graduations correctly.

In the following example, \texttt{numprint} uses the English conventions for writing numbers because I used:

\begin{tikzpicture}
\usepackage[english]{babel}
\end{tikzpicture}
5 Initialization \tkzInit
\begin{tikzpicture}
\tkzInit[xmin=0.00, xmax=0.05,  
ymin=1.2200,ymax=1.2215,  
xstep=0.01, ystep=0.0005]
\tkzAxeXY
\tkzDefPoint(.04,1.22025){I}
\tkzDrawPoint(I)
\end{tikzpicture}

5.2.3 Negative values

\begin{tikzpicture}
\tkzInit[xmin = -40,  
xmax = 60,  
ymin = -40,  
ymax = 60,  
xstep = 20,  
ystep = 20]
\tkzAxeXY
\end{tikzpicture}
6 Macros for the axes

Careful, these macros have been modified. It’s now easier to use the styles of Ti\textit{k}Z. \texttt{\textbackslash tkzDrawX} allows to draw an axis, \texttt{\textbackslash tkzLabelX} places graduations and finally in simple cases \texttt{\textbackslash tkzAxeX} traces and graduations. The options of Ti\textit{k}Z are accessible. Fractions can be used for graduations.

6.1 \texttt{\textbackslash tkzDrawX}

\begin{table}[h]
\begin{tabular}{|c|c|c|}
\hline
\texttt{options} & \texttt{default} & \texttt{definition} \\
\hline
\texttt{color} & black & Axis and ticks \\
\texttt{noticks} & false & no ticks on axis \\
\texttt{right space} & 0,5 cm & axis extended right \\
\texttt{left space} & 0 cm & extension of the axis to the left \\
\texttt{label} & \(x\) & label name \\
\texttt{trig} & 0 & if <>0 pi/trig is the unit \\
\texttt{tickwd} & 0,8pt & tick thickness \\
\texttt{tickup} & 1pt & tick over axis \\
\texttt{tickdn} & 1pt & tick depth over axis \\
\hline
\end{tabular}
\end{table}

This macro is used to draw the abscissa axis. The most important thing is to test all the options. Above, you have the values that define a tick. Otherwise the options of Ti\textit{k}Z apply and in particular \texttt{text, color, fill and font}.

6.1.1 No tick, no label

\begin{tikzpicture}
\tkzInit[xmax=5]
\tkzDrawX[label={},noticks]
\end{tikzpicture}

6.1.2 Label placement

\begin{tikzpicture}
\tkzInit[xmax=5]
\tkzDrawX[label = quantity, above left = 8pt]
\end{tikzpicture}
6.1.3 Label and Axis Colour

The color of the label is obtained with the option text, that of the axis with the option color. The option right=12pt shifts the label \( x \) by 12 pt.

\begin{tikzpicture}
\tkzInit[xmax=5]
\tkzDrawX[text=blue,color=red,right=12pt]
\end{tikzpicture}

6.1.4 Option right space

It adds a little space after the last tick.

\begin{tikzpicture}
\tkzInit[xmax=0.4,xstep=0.1]
\tkzDrawX[text=blue,color=red,right=12pt,right space=1]
\end{tikzpicture}

6.1.5 Trigonometric axis with the option trig=1

If number=0 then the axis is graduated from \( \text{cm} \) to \( \text{cm} \), otherwise the axis is graduated using multiples of \( \frac{s}{\text{number}} \).

\begin{tikzpicture}
\tkzInit[xmin=0,xmax=4,ymin=-1,ymax=1]
\tkzDrawX[trig=1]
\end{tikzpicture}

6.1.6 Trigonometric axis with the option trig=2

\begin{tikzpicture}
\tkzInit[xmin=0,xmax=4,ymin=-1,ymax=1]
\tkzDrawX[trig=2]
\end{tikzpicture}

6.2 \texttt{tkzLabelX[local options]}

This macro allows you to place graduations. The option orig can be used again, but its behavior is reversed. By default, the original value is placed. The options are those of Ti\texttt{kZ}, plus the following ones:

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>frac</td>
<td>0</td>
<td>if &lt;&gt;0 graduations = num/frac &quot;frac is an integer&quot;</td>
</tr>
<tr>
<td>trig</td>
<td>0</td>
<td>if &lt;&gt;0 pi/trig &quot;trig is an integer&quot;</td>
</tr>
<tr>
<td>font</td>
<td>\textstyle</td>
<td>scale size.</td>
</tr>
<tr>
<td>color</td>
<td>black</td>
<td>graduation color</td>
</tr>
<tr>
<td>step</td>
<td>1</td>
<td>interval between graduations</td>
</tr>
<tr>
<td>np</td>
<td>off=false</td>
<td>numprint deactivation</td>
</tr>
<tr>
<td>orig</td>
<td>true</td>
<td>displays the origin graduation</td>
</tr>
</tbody>
</table>

frac and trig are integers that can be changed to fractional or trigonometric writing.
6.2.1 Position of the graduations

\begin{tikzpicture}
\tkzInit[xmax=.5,xstep=0.1]
\tkzDrawX[label={$t$},text=blue,color=red]
\tkzLabelX[text=blue,below = 3pt]
\end{tikzpicture}

6.2.2 Position of the graduations with xlabel style

\begin{tikzpicture}
\tkzInit[xmin=1000,xmax=4000,xstep=1000]
\tkzDrawX
\tikzset{xlabel style/.append style={rotate=-30}}
\tkzLabelX[below right=3 pt,inner sep = 1pt]
\end{tikzpicture}

6.2.3 Dates with np off

For dates, you have to deactivate numprint.

\begin{tikzpicture}
\tkzInit[xmin=2000,xmax=2004]
\tkzDrawX
\tikzset{xlabel style/.append style={rotate=-30}}
\tkzLabelX[np off,below right=3 pt,inner sep =1pt]
\end{tikzpicture}

6.2.4 frac

\begin{tikzpicture}
\tkzInit[xmax=1.75,xstep=0.33333]
\tkzDrawX[label={$t$},text=blue,color=red]
\tkzLabelX[frac=3,text=blue,below = 6pt]
\end{tikzpicture}

6.2.5 trig

\begin{tikzpicture}
\tkzInit[xmin=0,xmax=5,ymin=-1,ymax=1]
\tkzDrawX[trig=2]
\tkzLabelX[trig=2,text=blue,below = 8pt]
\end{tikzpicture}

6.2.6 Graduations size

Two possibilities. It is possible to define the default style used for the math mode:

\let\tkzmathstyle\textstyle
6 Macros for the axes

6.2.7 Colour of the graduations

The key here is to use the color, text, and text options correctly. `fill`

6.2.8 Axis drawings before the graduation

In some cases, it is preferable to place \texttt{tkzDrawXY} after \texttt{tkzLabelX} and \texttt{tkzLabelY}.

This prevents display problems.
6.2.9 Graduations (except originally) prior to tracings

\begin{tikzpicture}
\tkzInit[xmin = -1, xmax = 4, ymin = -1, ymax = 1]
\tkzLabelX[orig=false]
\tkzLabelY[orig=false]
\tkzDrawXY
\end{tikzpicture}

6.2.10 Only positive graduations before drawings

\begin{tikzpicture}
\tkzInit[xmin=2, ymin=2, xmax=4, ymax=4]
\tkzLabelX \tkzLabelY
\tkzDrawXY
\end{tikzpicture}

6.2.11 No graduations at the origin

\begin{tikzpicture}
\tkzInit[xmin=2, ymin=2, xmax=4, ymax=4]
\tkzLabelX[orig] \tkzLabelY[orig]
\tkzDrawXY
\end{tikzpicture}

6.3 \tkzAxeX

\tkzAxeX[(local options)]

This macro allows you to draw the abscissa axis with default ticks as well as the graduations. It combines the two macros \tkzDrawX and \tkzLabelX. It should only be used in simple cases.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>x</td>
<td>label name</td>
</tr>
<tr>
<td>trig</td>
<td>0</td>
<td>graduation fraction de \pi</td>
</tr>
<tr>
<td>frac</td>
<td>0</td>
<td>fractional, denominator graduation \textless frac \textgreater</td>
</tr>
<tr>
<td>swap</td>
<td>false</td>
<td>allows you to run \tkzLabelX \textbefore \tkzDrawX</td>
</tr>
</tbody>
</table>

The option text defines the color of the graduations.
6.3.1 Example with \tkzAxeX

\begin{tikzpicture}
\tkzInit[xmax=0.5,xstep=0.1,ymax=1]
\tkzGrid
\tkzAxeX[text=blue]
\end{tikzpicture}

6.3.2 Use of pi and \tkzAxeX

\begin{tikzpicture}
\let\tkzmathstyle\displaystyle
\tkzLabelX[orig = false, frac = 4,below = 10pt]
\tkzDrawX[label = $t$]
\tkzAxeY[trig=2]
\end{tikzpicture}

6.3.3 Option frac and trig

In this example, we position the $t$ label as well as the graduations. \texttt{\below=10pt} is used to place the graduations underneath.

\begin{tikzpicture}
\tkzInit[xmax=9,xstep=3,ymax=3.5]
\tkzLabelX[below=10pt,orig=false,frac=3]
\tkzDrawX[label = $t$]
\tkzAxeY[trig=2]
\end{tikzpicture}

6.4 \tkzDrawY

\tkzDrawY[\{local options\}]

This macro allows you to draw the ordinate axis with default ticks. The options are those of Ti\textsc{k}Z plus the following ones:

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>color of axis and ticks</td>
</tr>
<tr>
<td>noticks</td>
<td>false</td>
<td>no ticks on the axis</td>
</tr>
<tr>
<td>up space</td>
<td>0,5 cm</td>
<td>top axis extension</td>
</tr>
<tr>
<td>down space</td>
<td>0 cm</td>
<td>axis extension down</td>
</tr>
<tr>
<td>label</td>
<td>$x$</td>
<td>label name</td>
</tr>
<tr>
<td>trig</td>
<td>0</td>
<td>if $&lt;&gt;0$ pi/trig is the unit</td>
</tr>
<tr>
<td>tickwd</td>
<td>0.8pt</td>
<td>tick’s thickness</td>
</tr>
<tr>
<td>ticklt</td>
<td>1pt</td>
<td>height of the tick above the axis</td>
</tr>
<tr>
<td>tickrt</td>
<td>1pt</td>
<td>above-axis tick depth</td>
</tr>
</tbody>
</table>

\texttt{tkz-base} AlterMundus
6 Macros for the axes

6.5 \tkzLabelY

\tkzLabelY\langle\text{local options}\rangle

This macro allows you to draw the abscissa axis with default ticks. The options are those of TikZ plus the following ones:

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>graduation color</td>
</tr>
<tr>
<td>frac</td>
<td>0</td>
<td>if (&lt;0) graduations are denominator fractions=frac</td>
</tr>
<tr>
<td>font</td>
<td>\textstyle</td>
<td>graduation size.</td>
</tr>
<tr>
<td>step</td>
<td>1</td>
<td>interval between graduations</td>
</tr>
</tbody>
</table>

frac and trig are integers that can be changed to fractional or trigonometric writing.

6.6 \tkzAxeY

\tkzAxeY\langle\text{local options}\rangle

This macro combines the two macros: \tkzDrawY \tkzLabelY Voir \tkzAxeX for options

6.7 \tkzAxeXY

\tkzAxeXY\langle\text{local options}\rangle

This macro combines the four macros: \tkzDrawX \tkzDrawY \tkzLabelX \tkzLabelY

It is necessary to use common options as in the example below, but this means that the same options are applied to both macros. Thus it is not possible to change label.

6.7.1 Colour of axes, graduations

\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=4,ymin=-1,ymax=3]
\tkzAxeXY[label={},text=blue]
\end{tikzpicture}
6.7.2 Option \texttt{label={}}

\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=4,ymin=-1,ymax=2]
\tkzAxeXY[label={},text=blue,trig=2]
\end{tikzpicture}

6.7.3 Option \texttt{swap}

\begin{tikzpicture}
\tkzInit[xmin=-2,xmax=2,ymin=-2,ymax=2]
\tkzAxeXY[label={},swap]
\end{tikzpicture}

6.8 \texttt{tkzDrawXY}

This macro combines the two macros: \texttt{tkzDrawX}\texttt{ tkzDrawY}. It is necessary to use common options as in the example below

6.8.1 Common colour and empty labels

\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=4,ymin=-1,ymax=1]
\tkzDrawXY[label={},color=red]
\end{tikzpicture}
6.8.2 Two trigonometric axes

\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=4,ymin=-1,ymax=1]
\tkzDrawXY[label={},color=red,trig=4]
\end{tikzpicture}

6.9 \tkzLabelXY

This macro combines the two macros:
\tkzLabelX\tkzLabelY
It is necessary to use common options as in the example below.

6.9.1

\begin{tikzpicture}
\tkzInit[xmin=-1,xmax=4,ymin=-1,ymax=1]
\tkzDrawXY[label={},color=red]
\tkzLabelXY[text=blue]
\end{tikzpicture}

6.10 Changing values by axis default

\begin{tabular}{|l|p{2cm}|p{10cm}|}
\hline
options & default & definition \\
\hline
line width & 0.4pt & line width defines the width of the line \\
tickwd & 0.8pt & tick thickness \\
ticka & 1pt & right side or above the tick \\
tickb & 1pt & left side or below the tick \\
font & \textstyle & graduation size. \\
\hline
\end{tabular}
6 Macros for the axes

6.10.1 Changing the default axes

You have to run \texttt{tkzSetUpAxis} again to retrieve the default values.

\begin{verbatim}
\begin{tikzpicture}[scale=1]
\tkzInit[ymax=2,xmax=4]
\tkzSetUpAxis[line width=1pt,tickwd=1pt,ticka=3pt, tickb=0pt]
\tkzAxeXY
\end{tikzpicture}
\end{verbatim}
A few changes for this macro. First of all, to simplify currently the color of the thinnest grid is determined automatically from the main grid, same process for the thickness. This behavior can be modified using styles.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x_A ; y_A))</td>
<td>((x_B ; y_B))</td>
<td>((xmin,ymin))(xmax,ymax) grid pattern</td>
</tr>
<tr>
<td>sub</td>
<td>true</td>
<td>asks for a sub-grid</td>
</tr>
<tr>
<td>color</td>
<td>darkgray</td>
<td>main grid color</td>
</tr>
<tr>
<td>subxstep</td>
<td>0.2</td>
<td>the step of the subgraduations for the abscissa axis</td>
</tr>
<tr>
<td>subystep</td>
<td>0.2</td>
<td>the step of the subgraduations for the ordinate axis</td>
</tr>
<tr>
<td>line width</td>
<td>0.4pt</td>
<td>main grid line thickness</td>
</tr>
</tbody>
</table>

Default values can be changed in the configuration file or by macros. The color of the second grid is the same as the main grid, but less intense. Same behavior for the line thickness. See the examples to change this behavior.

### 7.0.1 \texttt{tkzGrid} and the option \texttt{sub}

The option \texttt{sub} allows you to display a finer secondary grid. It is preferable to run \texttt{tkzGrid} first, to prevent the grid from being overlapped with other elements.

```latex
\begin{tikzpicture}
\tkzInit[xmax=4, ymax=2]
\tkzGrid[sub]
\tkzAxeXY
\end{tikzpicture}
```

### 7.0.2 Option \texttt{sub}

The option \texttt{sub} allows to display a finer secondary grid. Some parameters are mofiable.

```latex
\def\tkzCoeffSubColor{20}
\def\tkzCoeffSubLw{0.2}
\begin{tikzpicture}
\tkzInit[xmax=4, ymax=2]
\tkzGrid[sub, color=orange, subxstep=.5, subystep=.5]
\tkzAxeXY
\end{tikzpicture}
```
7 Use of \tkzGrid

7.0.3 Almost Default

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=2]
\tkzGrid[color=orange]
\tkzAxeXY
\end{tikzpicture}

7.0.4 Under the grid, too, option sub

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=2]
\tkzGrid[sub,color=orange]
\tkzGrid[color=orange]
\tkzAxeXY
\end{tikzpicture}

7.0.5 Grid change

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=2]
\tkzGrid[color = orange, sub, subxstep = 0.1, subystep = 0.1]
\tkzAxeXY
\end{tikzpicture}

7.0.6 Option xstep, xstep, subxstep and subystep

\begin{tikzpicture}
\tkzInit[xmax=.5,xstep=.1, ymax=.2,ystep=.1]
\tkzGrid[sub, subxstep = 0.05, subystep = 0.05, color=orange]
\tkzAxeXY
\end{tikzpicture}
7 Use of \texttt{\textbackslash tkzGrid}

7.0.7 With large intervals

\begin{tikzpicture}
\tkzInit[xmax=100,xstep=20, ymax=3000,ystep=1000]
\tkzGrid[sub, subxstep=10, subystep=500, color=orange]
\tkzAxeXY
\end{tikzpicture}

7.0.8 \texttt{\textbackslash tkzGrid} and the arguments

The grid can be any size.

\begin{tikzpicture}
\tkzInit[xmax=100,xstep=20, ymax=3000,ystep=1000]
\tkzGrid[sub, subxstep=10, subystep=500, color=orange]
\tkzAxeXY
\end{tikzpicture}

7.0.9 Use of \texttt{pi} with \texttt{\textbackslash tkzGrid}

\begin{tikzpicture}[scale=.75]
\tkzInit[xmax=6.5,ymax=6.5]
\tkzGrid[xstep=pi,ystep=pi/2,sub, subxstep=pi/4,subystep=pi/4]
\tkzLabelX[label=$\theta$,orig=false,trig=4, below=6pt,font=\scriptsize]
\tkzLabelY[trig=2,font=\scriptsize]
\tkzDrawXY
\end{tikzpicture}
7 Use of \tkzGrid

7.0.10 Options frac and trig with \tkzGrid

\begin{tikzpicture}
\tkzInit[xmax=9,xstep=3,ymax=4]
\tkzGrid[xstep=1,ystep=pi/2,sub,
subxstep=1,subystep=pi/4]
\tkzLabelX[label=$t$,orig=false,frac=3,
below=6pt,font=\scriptsize]
\tkzLabelY[trig=2,font=\scriptsize]
\end{tikzpicture}

7.0.11 Use of a repetition grid

\begin{tikzpicture}[scale=.5]
\tikzset{xaxe style/.style ={-}}
\tkzInit[xmax=15,ymax=15]
\tkzClip
\tkzGrid[sub,color=orange]
\tkzLabelX[label= ] \tkzLabelY[label= ]
\tkzDrawXY
\node[opacity=.5] at (8,6) {%
\includegraphics[scale=.5]{tiger}};
\end{tikzpicture}
8 The points

I made a distinction between the point used in Euclidean geometry and the point used to represent an element of a statistical cloud. In the first case, I use as object a node, which means that the representation of the point cannot be modified by a scale; in the second case, I use as object a plot mark. The latter can be scaled and have more varied forms than the node.

The new macro is \texttt{\texttt{\tkzDefPoint}}, it allows to use Ti\textsc{zk}-specific options as a shift and the values are processed with \texttt{tkz-base}. Moreover, if calculations are needed then the \texttt{xfp} package takes care of them. You can use Cartesian or polar coordinates.

8.1 Defining a point in Cartesian coordinates: \texttt{\tkzDefPoint}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{x,y}</td>
<td>no default</td>
<td>\texttt{x} and \texttt{y} are two dimensions, by default in cm.</td>
</tr>
<tr>
<td>\texttt{a:r}</td>
<td>no default</td>
<td>\texttt{a} is an angle in degrees, \texttt{r} is a dimension</td>
</tr>
</tbody>
</table>

The mandatory arguments of this macro are two dimensions expressed with decimals, in the first case they are two measures of length, in the second case they are a measure of length and the measure of an angle in degrees.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{shift}</td>
<td>(0,0)</td>
<td>value spacing</td>
</tr>
</tbody>
</table>

All the options of Ti\textsc{zk} that we can apply to \texttt{coordinate}, are applicable (well I hope!) as for example the option \texttt{label} defined with the library \texttt{quotes}.

8.1.1 Use of \texttt{shift}

\texttt{shift} allows the points to be placed in relation to each other.

\begin{tikzpicture}[trim left=-1cm]
\tkzDefPoint(2,3){A}
\tkzDefPoint[shift={(2,3)}](31:3){B}
\tkzDefPoint[shift={(2,3)}](158:3){C}
\tkzDrawSegments[color=red,line width=1pt](A,B A,C)
\tkzDrawPoints[color=red](A,B,C)
\end{tikzpicture}

8.2 Placing a label with the library \texttt{quotes}

I prefer not to mix operations and use \texttt{\tkzLabelPoint} to place labels. See section The Syntax Quotes.
8.2.1 Rotation with shift and scope

Preferable to rotate is to use an environment `scope`.

8.2.2 Forms and coordinates

Here we must follow the syntax of `xfp`. It is always possible to go through `pgfmath` but in this case, the coordinates must be calculated before using the macro `\tkzDefPoint`.

8.2.3 Scope et `\tkzDefPoint`

First, we can use the `scope` de Ti\k Z In the following example, we have a way to define an isosceles triangle.
8.3 Definition of points in Cartesian coordinates: \texttt{tkzDefPoints}

\begin{verbatim}
\tkzDefPoints{\langle\text{local options}\rangle\{(x_1/y_1/n_1,x_2/y_2/n_2, \ldots)\}}
\end{verbatim}

\(x_i\) et \(y_i\) are the coordinates of a referenced point \(n_i\)

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_i/y_i/n_i)</td>
<td>\texttt{tkzDefPoints{0/0,2/2/A}}</td>
</tr>
</tbody>
</table>

8.3.1 Definition of points

\begin{verbatim}
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,2/0/B,2/2/C,0/2/D}
\tkzDrawSegments(D,A A,B B,C C,D)
%with \texttt{tkz-euclide} \tkzDrawPolygon(A,...,D)
\tkzDrawPoints(A,B,C,D)
\end{tikzpicture}
\end{verbatim}

8.4 Point relative to another: \texttt{tkzDefShiftPoint}

\begin{verbatim}
\tkzDefShiftPoint[(\textit{Point})\{(x,y)\}\{\textit{name}\} ou (a:r)\{\textit{name}\}]
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x,y))</td>
<td>no default</td>
<td>(x) and (y) are two dimensions, by default in cm.</td>
</tr>
<tr>
<td>((a:r))</td>
<td>no default</td>
<td>(a) is an angle in degrees, (r) is a dimension</td>
</tr>
<tr>
<td>point</td>
<td>no default</td>
<td>\texttt{tkzDefShiftPoint{A}(0:4){B}}</td>
</tr>
</tbody>
</table>

No options. The name of the point is mandatory.

8.4.1 Example with \texttt{tkzDefShiftPoint}

This macro allows you to place one point relative to another. This is equivalent to a translation. Here is how to construct an isosceles triangle with main vertex \(A\) and angle at vertex of \(30^\circ\).
8.5 Point relative to another: \texttt{\textbackslash tkzDefShiftPointCoord}

\begin{verbatim}
\texttt{\textbackslash tkzDefShiftPointCoord[(a,b)]((x,y){(name)} \texttt{ou (a:r){(name)})}
\end{verbatim}

This involves performing a \((a,b)\) vector translation at the defined point relative to the origin.

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x,y))</td>
<td>no default</td>
<td>(x) and (y) are two dimensions, by default in cm.</td>
</tr>
<tr>
<td>((a:r))</td>
<td>no default</td>
<td>(a) is an angle in degrees, (r) is a dimension</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{a,b}</td>
<td>no default</td>
<td>\texttt{\textbackslash tkzDefShiftPointCoord<a href="">2,3</a>{B}}</td>
</tr>
</tbody>
</table>

The option is mandatory

8.5.1 Equilateral triangle with \texttt{\textbackslash tkzDefShiftPointCoord}

Let's see how to get an equilateral triangle (there is much simpler)

\begin{verbatim}
\begin{tikzpicture}[scale=1]
\texttt{\textbackslash tkzDefPoint(2,3){A}}
\texttt{\textbackslash tkzDefShiftPointCoord[2,3](30:4){B}}
\texttt{\textbackslash tkzDefShiftPointCoord[2,3](-30:4){C}}
\texttt{\textbackslash tkzDrawSegments(A,B,B,C,C,A)}
% or \texttt{\textbackslash tkzDrawPolygon}
\texttt{\textbackslash tkzDrawPoints(A,B,C)}
\texttt{\textbackslash tkzLabelPoints(A,B,C)}
\texttt{\textbackslash tkzLabelPoint[left](A){\$A\$}}
\end{tikzpicture}
\end{verbatim}

8.5.2 Isosceles triangle with \texttt{\textbackslash tkzDefShiftPointCoord}

Let's see how to obtain an isosceles triangle with a principal angle of 30 degrees. Rotation is possible. \(AB = AC = 5\) and \(BAC\)
\begin{tikzpicture}[rotate=15]
\tkzDefPoint(2,3){A}
\tkzDefShiftPointCoord[2,3](15:5){B}
\tkzDefShiftPointCoord[2,3](-15:5){C}
\tkzDrawSegments(A,B,B,C,C,A)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(B,C)
\tkzLabelPoint[left](A){$A$}
\end{tikzpicture}

8.6 Drawing a point \texttt{\tkzDrawPoint}

\begin{center}
\begin{tabular}{lll}
 arguments & default & definition \\
\hline
point & no default & a name or reference is requested \\
\hline
\end{tabular}
\end{center}

The argument is mandatory, but it is not necessary (although recommended) to use a reference; a pair of coordinates placed between braces is accepted. The disk takes the color of the circle, but 50\% lighter. It is possible to modify everything. The point is a node and is therefore invariant if the drawing is modified by scaling.

\begin{center}
\begin{tabular}{lll}
 options & default & definition \\
\hline
shape & circle & Possible \texttt{cross} or \texttt{cross out} \\
size & 2 pt & disk size \\
color & black & the default color can be changed \\
\hline
\end{tabular}
\end{center}

We can create other forms such as \texttt{cross}

8.6.1 Default stitch style

\begin{verbatim}
\begin{tikzpicture}
\tkzDefPoint(1,3){A}
\tkzDrawPoint(A)
\end{tikzpicture}
\end{verbatim}

8.6.2 Changing the style

The default definition in the file \texttt{tkz-base.cfg}

\begin{verbatim}
\tikzset{point style/.style={draw = \tkz@euc@pointcolor, inner sep = 0pt, shape = \tkz@euc@pointshape, minimum size = \tkz@euc@pointsise, fill = \tkz@euc@pointcolor!50}}
\end{verbatim}
8.6.3 Example of point plots

Note that scale does not affect the shape of the dots. Which is normal. Most of the time, we are satisfied with a single point shape that we can define from the beginning, either with a macro or by modifying a configuration file.

It is possible to draw several points at once, but this macro is a little slower than the previous one. Moreover, we have to make do with the same options for all the points.

8.7 Drawing points \tkzDrawPoints

\begin{tikzpicture}\[scale=.5\]
\tkzDefPoint(1,3){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(0,0){O}
\tkzDrawPoints[shape=cross out,size=12,color=red](A)
\tkzDrawPoints[shape=cross,size=12,color=blue](B)
\tkzDrawPoint[fill=green,size=12](O)
\tkzDrawPoint[size=12,color=blue,fill=yellow]({2,2})
\end{tikzpicture}

\begin{tikzpicture}[scale=.5]
\tkzDefPoint(1,3){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(0,0){O}
\tkzDrawPoints[shape=cross out,size=12,color=red](A)
\tkzDrawPoints[shape=cross,size=12,color=blue](B)
\tkzDrawPoint[size=12,color=green](O)
\tkzDrawPoint[size=12,color=blue,fill=yellow]({2,2})
\end{tikzpicture}

Warning at the final "s", an oversight leads to cascading errors if you attempt to plot multiple points. The options are the same as for the previous macro.

8.7.1 Example with \tkzDefPoint and \tkzDrawPoints

\begin{tikzpicture}
\tkzDefPoint(1,3){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(0,0){O}
\tkzDrawPoints[shape=cross out,size=12,color=red](A)
\tkzDrawPoints[shape=cross,size=12,color=blue](B)
\tkzDrawPoint[size=12,color=green](O)
\tkzDrawPoints[shape=cross,fill=yellow]({2,2})
\end{tikzpicture}

\begin{tikzpicture}[scale=.5]
\tkzDefPoint(1,3){A}
\tkzDefPoint(4,1){B}
\tkzDefPoint(0,0){O}
\tkzDrawPoints[size=8,color=red](A,B,O)
\end{tikzpicture}
8.7.2 More complex example

\begin{tikzpicture}[scale=.5]
\tkzDefPoint(2,3){A} \tkzDefPoint(5,-1){B}
\tkzDefPoint[label=below:$\mathcal{C}$, shift={(2,3)}](-30:5.5){E}
\begin{scope}[shift=(A)]
\tkzDefPoint(30:5){C}
\end{scope}
\tkzCalcLength[cm](A,B) \tkzGetLength{rAB}
\tkzDrawCircle[R](A,rAB cm)
\tkzDrawSegment(A,B)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(B,C)
\tkzLabelPoints[above](A)
\end{tikzpicture}

8.8 Add a label to a point \tkzLabelPoint

It is possible to add several labels at the same point by using this macro several times.

\begin{tabular}{|l|l|l|}
\hline
arguments & example & definition \\
\hline
point & \tkzLabelPoint(A){$A_1$} & \tkzLabelPoint \tkzLabelPoint \tkzLabelPoint \\
options & default & Tik\text{Z} options colour, position etc. \\
Tik\text{Z} options & & Optionally, we can use any style of Tikz, especially placement with above, right, dots... \\
\hline
\end{tabular}

8.8.1 Example with \tkzLabelPoint

\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B} \tkzDefPoint(0,3){C}
\tkzDrawSegments(A,B B,C C,A)
\tkzDrawPoints(A,B,C)
\tkzLabelPoint[left,red](A){$A$}
\tkzLabelPoint[right,blue](B){$B$}
\tkzLabelPoint[above,purple](C){$C$}
\end{tikzpicture}

8.8.2 Label and reference

The reference of a point is the object that allows to use the point, the label is the name of the point that will be displayed.
8.9 Add labels to points \texttt{\tkzLabelPoints}

It is possible to place several labels quickly when the point references are identical to the labels and when the labels are placed in the same way in relation to the points. By default, \texttt{below right} is chosen.

\begin{tabular}{l|l|l}
\textbf{arguments} & \textbf{example} & \textbf{result} \\
\hline
list of points & \texttt{\tkzLabelPoints(A,B,C)} & Display of \(A, B\) and \(C\) \\
\end{tabular}

This macro reduces the number of lines of code, but it is not obvious that all points need the same label positioning.

8.9.1 Example with \texttt{\tkzLabelPoints}

\begin{tikzpicture}
\tkzDefPoint(2,3){A}
\tkzDefShiftPoint[A](30:2){B}
\tkzDefShiftPoint[A](30:5){C}
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B,C)
\end{tikzpicture}

8.10 Automatic position of labels \texttt{\tkzAutoLabelPoints}

The label of a point is placed in a direction defined by a center and a point \texttt{center}. The distance to the point is determined by a percentage of the distance between the center and the point. This percentage is given by \texttt{dist}.

\begin{tabular}{l|l|l}
\textbf{arguments} & \textbf{example} & \textbf{result} \\
\hline
list of points & \texttt{\tkzAutoLabelPoints(A,B,C)} & Display of \(A, B\) and \(C\) \\
\end{tabular}

8.10.1 Example with \texttt{\tkzAutoLabelPoints}

Here the points are positioned relative to the center of gravity of \(A, B, C\) and \(O\).
8.10.2 Example with \tkzAutoLabelPoints

This time the reference is O and the distance is by default 0.15.

8.11 Point style with \tkzSetUpPoint

It is important to understand that the size of a dot depends on the size of a line.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape</td>
<td>circle</td>
<td>possible: circle, cross, cross out</td>
</tr>
<tr>
<td>size</td>
<td>current</td>
<td>the size of the point is size * line width</td>
</tr>
<tr>
<td>color</td>
<td>current</td>
<td>example \tkzLabelPoint(A,B,C)</td>
</tr>
<tr>
<td>fill</td>
<td>current!50</td>
<td>example \tkzLabelPoint(A,B,C)</td>
</tr>
</tbody>
</table>

This is a macro for choosing a style for points.
8.11.1 Simple example with \tkzSetUpPoint

\begin{tikzpicture}
\tkzSetUpPoint[shape = cross out, color=blue]
\tkzInit[xmax=100,xstep=20,ymax=.5]
\tkzDefPoint(20,1){A}
\tkzDefPoint(80,0){B}
\tkzDrawLine(A,B)
\tkzDrawPoints(A,B)
\end{tikzpicture}

8.11.2 Example with \tkzSetUpPoint

\begin{tikzpicture}
\tkzInit[ymin=-0.5,ymax=3,xmin=-0.5,xmax=7]
\tkzDefPoint(0,0){A}
\tkzDefPoint(0.25,0.25){B}
\tkzDefPoint(4,0){C}
\tkzDefPoint(3,2){D}
\tkzDrawSegments(A,B A,C A,D)
\tkzSetUpPoint[shape=cross out,size=4,]
\tkzDrawPoints(A,B,C,D)
\tkzLabelPoints(A,B,C,D)
\end{tikzpicture}

8.11.3 Using \tkzSetUpPoint in a group

Only the points in the group are affected by the changes.

\begin{tikzpicture}
\tkzInit[ymin=-0.5,ymax=3,xmin=-0.5,xmax=7]
\tkzDefPoint(0,0){A}
\tkzDefPoint(0.25,0.25){B}
\tkzDefPoint(4,0){C}
\tkzDefPoint(3,2){D}
\tkzDrawSegments(A,B A,C A,D)
{\tkzSetUpPoint[shape=cross out, fill= blue!70!black!!50, size=4, color=blue!70!black!30]}
\tkzDrawPoints(A,B)
\tkzSetUpPoint[fill= blue!70!black!!50, size=4, color=blue!70!black!30]
\tkzDrawPoints(C,D)
\tkzLabelPoints(A,B,C,D)
\end{tikzpicture}

8.12 Show point coordinates

This macro allows you to display the coordinates of a point and to draw arrows to specify the abscissa and ordinate. The point is given by its reference (its name). It is possible to give a couple of coordinates.
8 The points

\begin{center}
<table>
<thead>
<tr>
<th>argument</th>
<th>example</th>
<th>explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ref)</td>
<td>\tkzPointShowCoord(A)</td>
<td>shows the coordinates of point A</td>
</tr>
</tbody>
</table>

\begin{tabular}{|l|l|l|}
\hline
\textbf{option} & \textbf{default} & \textbf{explication} \\
\hline
xlabel & empty & label abscissa \\
xstyle & empty & style for the abscissa label node \\
oxdraw & false & boolean so as not to draw an arrow to (x') \\
\hline
\end{tabular}
\end{center}

8.12.1 Default styles

Here are some of the main styles:

\begin{verbatim}
\tikzset{arrow coord style/.style={dashed, \tkz@euc@linecolor, >=latex', ->}}
\tikzset{xcoord style/.style={\tkz@euc@labelcolor, font=\normalsize,text height=1ex, inner sep = 0pt, outer sep = 0pt, fill=\tkz@fillcolor, below=3pt}}
\tikzset{ycoord style/.style={\tkz@euc@labelcolor, font=\normalsize,text height=1ex, inner sep = 0pt, outer sep = 0pt, fill=\tkz@fillcolor, left=3pt}}
\end{verbatim}

8.12.2 Example with \texttt{\tkzPointShowCoord}

Without the options, you only get the arrows.

\begin{tikzpicture}[scale=1.5]
\tkzInit[xmax=3,ymax=2]
\tkzAxeXY
\tkzDefPoint(2,1){a}
\tkzPointShowCoord(a)
\tkzDrawPoint(a)
\tkzLabelPoint(a){$A_1$}
\tkzPointShowCoord(1,2)
\tkzDrawPoint(1,2)
\tkzLabelPoint(1,2){$A_2$}
\end{tikzpicture}

8.12.3 Example with \texttt{\tkzPointShowCoord} and \texttt{xstep}

Without the options, you only get the arrows.

\begin{tikzpicture}[scale=1.5]
\tkzInit[xmax=3,ymax=2]
\tkzAxeXY
\tkzDefPoint(2,1){a}
\tkzPointShowCoord(a)
\tkzDrawPoint(a)
\tkzLabelPoint(a){$A_1$}
\tkzPointShowCoord(1,2)
\tkzDrawPoint(1,2)
\tkzLabelPoint(1,2){$A_2$}
\end{tikzpicture}
8 The points

\begin{tikzpicture}[xscale=3,yscale=2]
\tkzInit[xmax=15,ymax=15, xstep=10,ystep=10]
\tkzAxeXY
\tkzDefPoint(10,10){a} \tkzDrawPoint(a)
\tkzPointShowCoord(a)
\tkzLabelPoint(a){$A_1$}
\end{tikzpicture}

8.13 \texttt{tkzDefSetOfPoints}

It was already possible to create a scatter plot with the macro \texttt{tkzDefPoints}, but this requires making a reference (a name) to each point, which is sometimes tedious. The macro \texttt{tkzSetOfPoints} allows to define points \texttt{tkzPt1}, \texttt{tkzPt2}, etc.

This is frequently referred to as « scatter plot ». The difference from the macro \texttt{tkzDefPoints} is that the reference to the points is given by a prefix (default \texttt{tkzPt}) and the point number. The points are not drawn.

\begin{verbatim}
\tkzDefSetOfPoints[⟨local options⟩]{⟨x₁/₁, y₁/₁, x₂/₂, …, xₙ/ₙ⟩}
\end{verbatim}

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{xₙ/yₙ}</td>
<td>no</td>
<td>default List of couples \texttt{xₙ/yₙ} separated by commas</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>prefix</td>
<td>\texttt{tkzPt}</td>
<td>prefix for point names</td>
</tr>
</tbody>
</table>

8.13.1 Creating a scatter plot with \texttt{tkzDefSetOfPoints}

\begin{tikzpicture}
\tkzInit[ymin=0,xmax=5]
\tkzAxeXY
\tkzDefSetOfPoints[prefix=P]%  \{1/2, 4/3, 2/2.5\}
\tkzDrawPoints(P1,P2,P3)
\tkzLabelPoints(P1,P2,P3)
\end{tikzpicture}
9 Style Use

9.1 Modification of tkz-base

tkz-base.sty has a default configuration file. Its existence is not mandatory, but if it exists, you can modify it to get different default styles. I only give a quick description of this file, as it may evolve soon.

In tkz-base.cfg, you can set the axes, the reference (if used), the grid, etc. as well as the styles which are linked to these objects. It is possible to modify the styles of the points and segments.

It is also possible to define the dimensions of a drawing by default by modifying xmin, xmax, ymin and ymax.

\def\tkz@xa{0}
\def\tkz@xb{10}
\def\tkz@ya{0}
\def\tkz@yb{10}

These lines are used to define the values of xmin, xmax, etc.

You can change them, for example:

\def\tkz@xa{-5}
\def\tkz@xb{-5}
\def\tkz@ya{5}
\def\tkz@yb{5}

Here’s a list of used styles you’ll find in tkz-base.cfg

- xlabel style
- xaxe style
- ylabel style
- yaxe style
- rep style
- line style
- point style
- mark style
- compass style
- vector style
- arrow coord style
- xcoord style
- ycoord style

9.2 Use \tikzset

It’s better to use \tikzset now rather than \tikzstyle and it’s possible to use tkz-base.cfg.

If you want to change the appearance of the axes of the orthogonal coordinate system, for example place arrows at each end or remove them. This can be done in tkz-base.cfg or in your code.
The transformation will be valid for the entire document. Note that $\texttt{xmin}$ has been modified, in fact the arrow and the line corresponding to the graduation merge.

\begin{tikzpicture}
\tkzInit[xmin=-0.5,xmax=5]
\tkzDrawX
\tkzLabelX
\end{tikzpicture}

9.3 Configuration Macros

- \texttt{\tkzSetUpPoint}
- \texttt{\tkzSetUpAxis}
10 Bounding box management

The initial bounding box after using the macro \texttt{\texttt{tkzInit}} is defined by the rectangle based on the points (0,0) and (10,10). The \texttt{\texttt{tkzInit}} macro allows this initial bounding box to be modified using the arguments (\texttt{xmin}, \texttt{ xmax}, \texttt{ ymin}, and \texttt{ ymax}). Of course any external trace modifies the bounding box. \texttt{Ti\k Z} maintains that bounding box. It is possible to influence this behavior either directly with commands or options in \texttt{Ti\k Z} such as a command like \texttt{\texttt{useasboundingbox}} or the option \texttt{use as bounding box}. A possible consequence is to reserve a box for a figure but the figure may overflow the box and spread over the main text. The following command \texttt{\texttt{pgfresetboundingbox}} clears a bounding box and establishes a new one.

10.1 \texttt{tkzShowBB}

The simplest macro.

\texttt{\texttt{tkzShowBB}}[\texttt{local options}]

This macro displays the bounding box. A rectangular frame surrounds the bounding box. This macro accepts \texttt{Ti\k Z} options.

10.1.1 Example with \texttt{tkzShowBB}

\begin{tikzpicture}[scale=.5]
\tkzInit[ymax=5,xmax=8]
\tkzGrid
\tkzDefPoint(3,0){A}
\begin{scope}
\tkzClipBB
\tkzDrawCircle[R](A,5 cm)
\tkzShowBB
\end{scope}
\tkzDrawCircle[R,red](A,4 cm)
\end{tikzpicture}

10.2 \texttt{tkzClipBB}

\texttt{tkzClipBB}

The idea is to limit future constructions to the current bounding box.
10.2.1 Example with `\tkzClipBB` and the bisectors

\begin{tikzpicture}
\tkzInit[xmin=-3,xmax=6, ymin=-1,ymax=6]
\tkzDefPoint(0,0){O}\tkzDefPoint(3,1){I}\tkzDefPoint(1,4){J}
\tkzDefLine[bisector](I,O,J) \tkzGetPoint{i}
\tkzDefLine[bisector out](I,O,J) \tkzGetPoint{j}
\tkzDrawPoints(O,I,J,i,j)
\tkzClipBB\tkzDrawLines[add = 1 and 2,color=red](O,I O,J)
\tkzDrawLines[add = 1 and 2,color=blue](O,i O,j)
\tkzShowBB\end{tikzpicture}

10.3 `\tkzSetBB`

\begin{tikzpicture}
\tkzDefPoint(0,0){A}\tkzDefPoint(3,3){B}\tkzDefPoint(1,1){C}
\tkzSetBB(A)(2,2)\tkzDrawSegment(A,B)\tkzDrawPoints(A,C)\tkzShowBB\end{tikzpicture}

10.4 `\tkzSaveBB`

\begin{tikzpicture}
\begin{scope}
\tkzSetBB(0,0)(6,2)\tkzShowBB[fill=blue!20]\tkzSaveBB\tkzDefPoint(3,3){A}\tkzShowBB\tkzDrawCircle[R,fill=yellow,opacity=.2](A,2cm)\tkzRestoreBB\end{scope}\end{tikzpicture}

This macro defines the rectangle with coordinates \((x_A; y_A)\) and \((x_B; y_B)\) as the new bounding box.

This macro saves the bounding box, i.e. it stores the coordinates of two points that define a rectangle.
10 Bounding box management

10.5 \texttt{tkzRestoreBB}

\begin{verbatim}
\tkzRestoreBB
\end{verbatim}

This macro retrieves the bounding box backup. As you can see, the figure overflows the box. The bounding box has been reduced.

10.5.1 Example of the use of \texttt{tkzRestoreBB}

\begin{verbatim}
\vspace{2cm}
\begin{tikzpicture}
\tkzDefPoint(-2,-2){A}
\tkzDefPoint(2,1){B}
\tkzDefPoint(0,0){O}
\tkzSaveBB
\tkzShowBB[red,line width=1pt]
\tkzRestoreBB
\tkzDrawCircle(O,B)
\tkzClipBB
\tkzFillCircle[gray!20](O,B)
\end{tikzpicture}
\end{verbatim}

10.6 \texttt{tkzClip}

\begin{verbatim}
\tkzClip[⟨local options⟩]
\end{verbatim}

The role of this macro is to make invisible what is outside the rectangle defined by \((xmin; ymin)\) and \((xmax; ymax)\).

\begin{tabular}{l c c}
\hline
options & default & definition \\
\hline
space & 1 & added value on the right, left, bottom and top of the background \\
\hline
\end{tabular}

The role of the \texttt{space} option is to enlarge the visible part of the drawing. This part becomes the rectangle defined by \((xmin - space; ymin - space)\) and \((xmax + space; ymax + space)\). \texttt{space} can be negative! The unit is cm and should not be specified.
10.6.1 First example with \tkzClip

\begin{tikzpicture}
\tkzInit[xmax=3, ymax=3]
\tkzGrid
\tkzAxeXY
\draw[red] (-1,-1)--(5,5);
\end{tikzpicture}

It is possible to add a bit of space \tkzClip[space].

10.6.2 Second example with \tkzClip

\begin{tikzpicture}
\tkzInit[xmax=3, ymax=3]
\tkzGrid
\tkzAxeXY
\tkzClip
\draw[red] (-1,-1)--(5,5);
\end{tikzpicture}

The dimensions to define the clipped rectangle are xmin-1, ymin-1, xmax+1 and ymax+1.

10.6.3 \tkzClip et l'option space

\begin{tikzpicture}
\tkzInit[xmax=3, ymax=3]
\tkzGrid \tkzAxeXY
\tkzClip[space=-0.5]
\draw[red] (-0.5,-0.5)--(3.5,3.5);
\end{tikzpicture}

10.7 Reverse clip: tkzreverseclip

The next example uses
10.7.1 Example with \tkzClipOutPolygon

\begin{tikzpicture}[scale=.5]
\tkzInit[xmin=-5,xmax=5,ymin=-5,ymax=5]
\pgfinterruptboundingbox
\tkzDefPoints{-.5/0/P1,.5/0/P2}
\foreach \i [count=\j from 3] in {2,...,7}{%
    \tkzDefShiftPoint[P\i]({45*(\i-1)}:1 cm){P\j}
}\endpgfinterruptboundingbox
\tkzClipOutPolygon(P1,P2,P3,P4,P5,P6,P7,P8)
\tkzCalcLength[cm](P1,P5)\tkzGetLength{r}
\begin{scope}[blend group=screen]
\foreach \i in {1,...,8}{%
    \pgfmathparse{100-5*\i}
    \tkzFillCircle[R,color=blue!\pgfmathresult](P\i,\r)
}\end{scope}
\end{tikzpicture}

10.8 Options from TikZ: trim left or right

See the \texttt{pgfmanual}

10.9 TikZ Controls \texttt{\pgfinterruptboundingbox} and \texttt{\endpgfinterruptboundingbox}

This command temporarily interrupts the calculation of the box and configures a new box.

\begin{tikzpicture}
\tkzDefPoint(0,5){A}\tkzDefPoint(5,4){B}
\tkzDefPoint(0,0){C}\tkzDefPoint(5,1){D}
\pgfinterruptboundingbox
\tkzInterLL(A,B)(C,D)\tkzGetPoint{I}
\endpgfinterruptboundingbox
\tkzClipBB
\tkzDrawCircle(I,B)
\tkzDrawSegments(A,B C,D A,C)
\end{tikzpicture}
11 Use Additional Objects or Tools

These complementary objects can be particular points, straight lines, circles, arcs, etc. Now \texttt{tkz-base} has been minimized. If you want to use particular objects you must use \texttt{tkz-euclide}.
12 Using an orthogonal coordinate system

12.1 Repère avec \texttt{\tkzRep}

\begin{tabular}{l l l}
\hline
\texttt{\tkzRep[\{local options\}]} & options & default & definition \\
\hline
line width & 0.8pt & line width defines the width of the line \\
xlabel & \rightarrow & label for the abscissa axis \\
ylabel & \rightarrow & label for the ordinate axis \\
posxlabel & below=2pt & Label position \\
posylabel & left=2pt & Label position \\
xnorm & 1 & norm of the x-vector \\
ynorm & 1 & vector norm in y \\
color & black & line colour \\
colorlabel & black & label color \\
\hline
\end{tabular}

12.1.1 Some modifiable styles

\begin{verbatim}
\tikzset{xlabel style/.style = {below = 3 pt, inner sep = 1pt, outer sep = 0pt}}
\tikzset{ylabel style/.style = {left = 3 pt, inner sep = 1pt, outer sep = 0pt}}
\tikzset{xaxe style/.style = {> = latex, ->}}
\tikzset{yaxe style/.style = {> = latex, ->}}
\end{verbatim}

12.1.2 Example of use

\begin{verbatim}
\begin{tikzpicture}
\tikzset{xaxe style/.style={-}}
\tikzset{yaxe style/.style={-}}
\tkzInit[xmax=4,ymax=4]
\tkzGrid
\tkzDrawX
\tkzDrawY
\tkzRep[color=red,ynorm=2]
\end{tikzpicture}
\end{verbatim}

For those who use \texttt{french} with \texttt{babel}, in case of problems with version 3 of \texttt{pgf}, just load the \texttt{babel} library. \texttt{TikZ} was indeed sometimes allergic to the active characters.
13 Lines parallel to the axes

13.1 Draw a horizontal line with \texttt{tkzHLine}

\begin{itemize}
\item The syntax is that of \texttt{xfp}!
\end{itemize}

\begin{itemize}
\item \texttt{\textbackslash tkzHLine[\{local options\}][\{decimal number\}]}
\end{itemize}

\begin{itemize}
\item \texttt{decimal number} \quad \texttt{\textbackslash tkzHLine\{1\}} \quad \texttt{Draw the right} \quad \texttt{\textit{y = 1}}
\end{itemize}

\begin{itemize}
\item \texttt{options} \quad \texttt{default} \quad \texttt{definition}
\item \texttt{color} \quad \texttt{black} \quad \texttt{line colour}
\item \texttt{line width} \quad \texttt{0.6pt} \quad \texttt{point thickness}
\item \texttt{style} \quad \texttt{solid} \quad \texttt{line style}
\end{itemize}

\texttt{see options the lines in TikZ}

13.1.1 Horizontal line

\begin{tikzpicture}[scale=2]
\tkzInit[xmax=3,ymax=1.5]
\tkzAxeXY
\tkzHLine[color = blue, style = dashed, line width = 2pt]{1}
\end{tikzpicture}

13.1.2 Horizontal line and value calculated by xfp

\begin{tikzpicture}
\tkzInit[xmin=-3,xmax=3,ymin=-2,ymax=1.5]
\foreach\v in {-1,1}
\tkzHLine[color=red]{\v*pi/2}
\tkzDrawY
\tkzAxeX[trig=2]
\tkzLabelY
\end{tikzpicture}
13 Lines parallel to the axes

13.2 Horizontal lines with \tkzHLines

\[ \text{The syntax is that of } \textbf{xfp!} \]

\[
\textbf{\textbackslash tkzHLines}\{\text{local options}\}\{\text{list of values}\}
\]

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>list of values</td>
<td>\tkzHLines{1,4}</td>
<td>draws the lines ( x = 1 ) and ( x = 4 )</td>
</tr>
</tbody>
</table>

13.2.1 Horizontal lines

\[
\begin{tikzpicture}
\tkzInit[xmax=5,ymax=4]
\tkzAxeXY
\tkzHLines[color = magenta]{1,...,3}
\end{tikzpicture}
\]

13.3 Draw a vertical line with \tkzVLine

\[ \text{The syntax is that of } \textbf{xfp!} \]

\[
\textbf{\textbackslash tkzVLine}\{\text{local options}\}\{\text{decimal number}\}
\]

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal number</td>
<td>\tkzVLine{1}</td>
<td>Draw the line ( x = 1 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>line colour</td>
</tr>
<tr>
<td>line width</td>
<td>0.6pt</td>
<td>point thickness</td>
</tr>
<tr>
<td>style</td>
<td>solid</td>
<td>line style</td>
</tr>
</tbody>
</table>

See options the lines in \textit{TikZ}. 
13.3.1 Vertical line

\begin{tikzpicture}[scale=2]
\tkzInit[xmax=3,ymax=1]
\tkzAxeXY
\tkzVLine[color = blue, style = dashed, line width = 2pt]{1/3}
\end{tikzpicture}

13.3.2 Vertical line and value calculated by xfp

\begin{tikzpicture}
\tkzInit[xmax=7,ymin=-1,ymax=1]
\foreach\v in {1,2}{\tkzVLine[color=red]{\v*pi}}
\tkzDrawY
\tkzAxeX[trig=2]
\tkzLabelY
\end{tikzpicture}

13.4 Vertical lines with \tkzVLines

\begin{itemize}
  \item The syntax is that of xfp!
\end{itemize}

\begin{tabular}{|c|c|c|}
  \hline
  \textbf{arguments} & \textbf{example} & \textbf{definition} \\
  \hline
  list of values & \tkzVLines{1,4} & Trace the lines $x=1$ and $x=4$ \\
  \hline
\end{tabular}

13.4.1 Vertical lines

\begin{tikzpicture}
\tkzInit[xmax=5,ymax=2]
\tkzAxeXY
\tkzVLines[color = green]{1,2,...,4}
\end{tikzpicture}

14 Ticks on the axes

14.1 Drawing one tick on the abscissa axis \tkzHTick
14 Ticks on the axes

\tkzHTick[[local options]]{(decimal number)}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal number</td>
<td>\tkzHTick{1}</td>
<td>the abscissa of the tick is 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mark</td>
<td>*</td>
<td>full disk</td>
</tr>
<tr>
<td>mark size</td>
<td>3 pt</td>
<td>symbol size</td>
</tr>
<tr>
<td>mark options</td>
<td>empty</td>
<td>allows you to use color for example</td>
</tr>
</tbody>
</table>

See options for Ti\kZ.  

14.1.1 Example

\begin{tikzpicture}
\tkzInit[xmax=6]
\tkzDrawX
\tkzHTick[mark=ball,mark size=3pt]{\pi/2}
\tkzHTick[mark=*,mark options={color=purple}]{2*exp(1)}
\end{tikzpicture}

14.2 Drawing ticks on the ordinate axis \tkzHTicks

\tkzHTicks[[local options]]{(list of numbers)}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal number</td>
<td>\tkzHTicks{1}</td>
<td>the abscissa of the tick is 1</td>
</tr>
</tbody>
</table>

See options for Ti\kZ.  

14.3 Drawing one tick on the abscissa axis \tkzVTick

\tkzVTick[[local options]]{(decimal number)}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal number</td>
<td>\tkzVTick{1}</td>
<td>the abscissa of the tick is 1</td>
</tr>
</tbody>
</table>

See options for Ti\kZ.  

14.4 Drawing ticks on the abscissa axis \tkzVTicks

\tkz-base AlterMundus
\texttt{\textbackslash tkzVTicks[(local_options)]\{decimal\ number\}}

<table>
<thead>
<tr>
<th>arguments</th>
<th>example</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal number</td>
<td>\texttt{tkzVTicks1,3}</td>
<td>the ordinates of the ticks are 1 and 3</td>
</tr>
</tbody>
</table>

See options for \texttt{TikZ}.
I distinguished between the points used in Euclidean geometry and the "marks or symbols" that can be found in statistics.

To position the symbol, we use the macro \texttt{\tkzDefPoint} to correctly define a point, then the macro \texttt{\tkzDrawMark} to draw the symbol.

It is common to have to draw a scatter plot, so I created a macro that allows you to define several points quickly. A "mark" symbol can be scaled, which is sometimes useful, but, on the other hand, if you change the abscissa and ordinates differently then the "marks" are distorted.

Reminder: it was already possible to create a cloud of points with the macro \texttt{\tkzDefPoints}, but this requires to give a reference (a name) to each point, which is sometimes tedious. The macro \texttt{\tkzSetOfPoints} allows to define points \texttt{tkzPt1}, \texttt{tkzPt2}, etc.

This is frequently referred to as the "scatter plot". The difference from the macro \texttt{\tkzDefPoints} is that the reference to the points is given by a prefix (default \texttt{tkzPt}) and the point number.

The points are not drawn.

\subsection{15.1 \texttt{\tkzDrawSetOfPoints}}

\begin{verbatim}
\tkzDrawSetOfPoints[\langle local options \rangle]
\end{verbatim}

Allows you to place symbols on the points defined by \texttt{\tkzDefSetOfPoints}.

\begin{itemize}
\item options default definition
\item prefix \texttt{tkzPt} point name prefix
\end{itemize}

\subsubsection{15.1.1 Drawing of a scatter plot with \texttt{\tkzDrawSetOfPoints}}

\begin{tikzpicture}[scale=0.75]
\tkzInit[xmax=6,ymin=1000,ymax=5000,ystep=1000]
\tkzDrawX[label=$m$,below=10pt]
\tkzDrawY[label=$R(m)$,above=10pt]
\tkzLabelX[font=\scriptsize]
\tkzLabelY[font=\scriptsize]
\tkzDefSetOfPoints[show]{1/2000,2/3000,4/2500,5/4200}
\tkzDrawSetOfPoints[mark=ball,mark size=3pt]
\end{tikzpicture}

\subsection{15.2 \texttt{\tkzJoinSetOfPoints}}

\begin{verbatim}
\tkzJoinSetOfPoints[\langle local options \rangle]
\end{verbatim}

Allows the symbols to be joined by line segments. Of course, it is possible to use all the options of TikZ.

\begin{itemize}
\item options default definition
\item prefix \texttt{tkzPt} point name prefix
\end{itemize}
15.2.1 Link the points of a scatter plot with \texttt{tkzJoinSetOfPoints}

\begin{tikzpicture}[scale=1]
\tkzInit[xmax=5, ymin=1000,ymax=6000,ystep=1000]
\tkzDrawX[label=$m$,below=13pt]
\tkzDrawY[label=$R(m)$]
\tkzLabelX[font=\scriptsize]
\tkzLabelY[font=\scriptsize]
\tkzDefSetOfPoints{1/2000,2/3000,4/2500,5/4200}
\tkzJoinSetOfPoints[thick, color=brown]
\tkzDrawSetOfPoints[mark=ball, mark size=3pt]
\end{tikzpicture}

15.2.2 Using the points of a scatter plot

\begin{tikzpicture}[scale=.5]
\tkzInit[xmax=5,ymin=1000,ymax=6000,ystep=1000]
\tkzGrid[color=orange!30]
\tkzDrawX[label=$m$,below=13pt]
\tkzDrawY[label=$R(m)$]
\tkzLabelX[font=\scriptsize]
\tkzLabelY[font=\scriptsize]
\tkzDrawPolySeg[color=brown!50, line width=2pt](P1,P2,P3,P4,P5)
\end{tikzpicture}

15.3 \texttt{tkzSetUpMark}

\begin{table}[h]
\begin{tabular}{|c|c|c|}
\hline
\texttt{tkzSetUpMark}[\texttt{(local options)}] & options & default & example \\
\hline
\texttt{liste} & no default & \texttt{\tkzSetUpMark[mark=heart]} \\
\hline
\end{tabular}
\end{table}
15 Marks or symbols

15.3.1 Two scatter plots

\begin{tikzpicture}
\tkzInit[xmax=5.5,ymax=6000,\% ymax=6000,ystep=2000]
\tkzGrid[color=orange!30]
\tkzDrawX[label=\$m\$,below=13pt]
\tkzDrawY[above left,label=\$R(m)\$]
\tkzLabelX[below right,font=\scriptsize]
\tkzLabelY[font=\scriptsize]
\tkzDefSetOfPoints{1/2000,2/3000,3/2000,4/2500,5/4200}
\tkzDefSetOfPoints[prefix=P]{1/3200,2/4100,3/3300,4/3300,5/5000}
\tkzSetUpMark[mark=heart,color=black,fill=red!30,size=4pt]
\tkzJoinSetOfPoints[thick,color=blue,double]
\tkzDrawSetOfPoints[thick,color=orange]
\tkzJoinSetOfPoints[prefix=P,mark=square*,mark size=4pt,mark options={color=blue,fill=blue!40}]
\tkzText[draw,color = red,fill = orange!20](3,5800){Recette en fonction du mois}
\end{tikzpicture}

15.4 \texttt{tkzDrawMark}

\begin{tikzpicture}
\tkzInit[xmax=3,ymax=1]
\tkzAxeXY
\tkzDrawMark[mark=ball](1,.5)
\end{tikzpicture}

15.4.1 Ball; use of \texttt{tkzDrawMarks}

\begin{tikzpicture}
\tkzInit[xmax=3,ymax=1]
\tkzAxeXY
\tkzDrawMark[mark=ball](1,.5)
\end{tikzpicture}

15.5 \texttt{tkzDrawMarks}

\begin{tikzpicture}
\tkzInit[xmax=3,ymax=1]
\tkzAxeXY
\tkzDrawMark[mark=ball](1,.5)
\end{tikzpicture}
15 Marks or symbols

15.5.1 Mark and plot; use of \texttt{\textbackslash tkzDrawMarks}

\begin{tikzpicture}
\tkzInit[xmax=6,ymin=1000,ymax=5000,ystep=1000]
\tkzAxeXY
\tkzDefSetOfPoints[prefix=P]{{1/2000, 2/3000, 4/2500, 5/4200}}
\tkzDrawSegments[color=brown!50][P1,P2 P2,P3 P3,P4]
\tkzDrawMarks[mark=ball](P1,P2,P3,P4)
\end{tikzpicture}
16 Texts and Legends

16.1 Placing a title

Of course you can use TikZ, but the macro I propose to allow you to place the text using the units chosen for the drawing.

The options are always those of TikZ, in particular the following ones:

\[
\text{\texttt{\textbackslash tkzText[\langle local \ options \rangle]\langle \text{dot} \rangle\langle \text{text} \rangle}}
\]

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>black</td>
<td>current colour</td>
</tr>
<tr>
<td>text</td>
<td>black</td>
<td>text colour</td>
</tr>
<tr>
<td>fill</td>
<td>white</td>
<td>background colour</td>
</tr>
<tr>
<td>opacity</td>
<td>1</td>
<td>opacity</td>
</tr>
</tbody>
</table>

The point can either be given by its coordinates or by its name.

16.1.1 A title

\begin{tikzpicture}
\tkzInit[xmax = 6, ymin = 1000, ymax = 4000,ystep = 1000]
\tkzAxeXY
\tkzText[draw, line width = 1pt, color = red, fill = orange!20](3,4000){Recette en fonction du mois}
\end{tikzpicture}

16.1.2 Draft

\begin{tikzpicture}
\tkzInit[xmax = 6, ymin = 1000, ymax = 4000,ystep = 1000]
\tkzGrid \tkzAxeXY
\tkzText[draw,opacity=.2, rotate=45,inner sep=.6 cm, line width = 1pt, color = black, fill = purple!20](3,2500){\Huge DRAFT}
\end{tikzpicture}

16.1.3 Text with a period

It is possible to give the reference of a point instead of its coordinates.
16.1.4 Text format

The option `text width` is interesting, see the pgfmanual for more information.

\begin{tikzpicture}
  \tkzInit[ymax=5,xmax=6]
  \tkzAxeXY
  \tkzDefPoint(3,3){A}
  \tkzText[draw,opacity=.6, inner sep=.6 cm, line width = 1pt, color = black, fill = purple!20](A)
  \node[fill = purple!20, text width=3cm] at (3,3) {My text \ Reference};
\end{tikzpicture}

16.2 Placing legends

There are two ways to use this macro. Either you can place legends for curves. Then you can represent lines with their own style, or you can differentiate symbols (mark).

\begin{tikzpicture}
  \tkzInit[ymax=5,xmax=6]
  \tkzAxeXY
  \tkzDefPoint(3,3){A}
  \tkzText[draw,opacity=.6, inner sep=.6 cm, line width = 1pt, color = black, fill = purple!20](A)
  \node[fill = purple!20, text width=3cm] at (3,3) {My text \ Reference};
\end{tikzpicture}

The arguments differ according to the boolean `line`.

<table>
<thead>
<tr>
<th>options</th>
<th>default</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>false</td>
<td>Boolean : line or symbol</td>
</tr>
</tbody>
</table>

With `line=true`

<table>
<thead>
<tr>
<th>arguments</th>
<th>default</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>style/line width/color/texte</td>
<td>no default</td>
<td>dashed/1pt/red/Recipe</td>
</tr>
</tbody>
</table>

With `line=false`
16.2.1 Legends with symbols

\begin{tikzpicture}
\tkzInit[xmax=12,ymin=1000,ymax=11000,ystep=2000]
\tkzGrid[color=orange!30]
\tkzDrawX[below right,label=Mois]
\tkzDrawY[above left,label=Recette]
\tkzLabelX
\tkzLabelY
\tkzDefSetOfPoints{1/2000,2/3000,3/2000,4/2500,5/4200,6/4800,7/4600,8/5200,9/6200,10/7000,11/7400,12/10000}
\tkzDefSetOfPoints[prefix=P]{1/3200,2/4100,3/3300,4/3300,5/5000,6/5500,7/5200,8/4000,9/3000,10/6000,11/8400,12/9000}
\tkzSetUpMark[mark=heart,color=black,fill=red!30,size=4pt]
\tkzJoinSetOfPoints[thick,color=brown,double]
\tkzDrawSetOfPoints[prefix=P,mark=square*,mark size=4pt,mark options={color=blue,fill=blue!40}]
\tkzLegend[draw,rounded corners,fill=orange!20,text=brown,line width=2pt](5,10000){\textit{heart}/1ex/red!30/Product Recipe A,\%
\textit{square}/0.75ex/blue!40/Product Recipe B}
\end{tikzpicture}
17 FAQ

17.1 General Questions

– **Why **\texttt{tkz-base}? As a Mathematics teacher, I needed tools that would allow me to write my lessons and exercises quickly. Ti\textsc{k}Z was perfect for that, but I was wasting too much time on details. I wanted to create a syntax that was both close to that of \textsc{B}i\textsc{I}X\textsc{E}X and \textsc{math} so I could memorize better. So I created a module for each branch of mathematics I taught. \texttt{tkz-base} is the common part of all these modules. \texttt{tkz-euclide} and \texttt{tkz-berge} are the ones I invested the most in.

– **Relationship with Ti\textsc{k}Z?** Ti\textsc{k}Z is a great package for describing drawings. My packages are based on it. That said, it is in no way comparable. My packages are only useful for people who want to create mathematical figures.

17.2 Most common errors

– **Error unknown option: ”label options”.** This option is no longer available. You can now directly use the options in Ti\textsc{k}Z.

– **Error avec \texttt{tkzDrawPoint ou \texttt{tkzDefPoint \texttt{tkzDrawPoint}(A,B)} when you need \texttt{tkzDrawPoints}.** This is true with all macros that allow you to define multiple objects. The singular form allows you to use custom options. On the other hand, it is possible to use the plural form for a single object.

– **Propagation of a style** It is possible to restrict the propagation of a style by placing a piece of code in a group or in a \texttt{scope} environment or between parentheses.

– **The use of the comma** even in a Mathematical mode $2.5$ needs to be protected in a \textsc{TeX} group, for example \{$2,5$}.

– \texttt{tkzDrawSegments{B,B' C,C'}} is a mistake. Only macros that define an object use braces.

– If an error occurs in a calculation when passing parameters, then it is better to make these calculations before calling the macro.

– Do not mix the syntax of \texttt{pgfmath} and \texttt{xfp}.
Index
\\below=10pt, 20
\\endpgfinterruptboundingbox, 47
Environment scope, 30
Operating System Windows, 8
Package autolanguage, 9
fp, 7, 9
numprint, 9
pgf, 7, 9
pgfmath, 30, 62
siunitx, 9
tkz-base, 8–10, 62
tkz-berge, 62
tkz-euclide, 7, 9, 62
tkz-fct, 7, 9, 10
xfp, 7, 9, 29, 30, 62
\\pgfinterruptboundingbox, 47
\\pgfresetboundingbox, 43
TeX Distributions
MikTeX, 8
TeXLive, 8
\\textstyle, 23
TkZ Library
- angles, 7
  babel, 9
  quotes, 7
\\tkzset, 41
\\tkzstyle, 41
\\tkzAutoLabelPoints, 36, 37
\\tkz Axe X, 15, 19–21
\\tkz Axe X: options
  frac, 19
  label, 19
  swap, 19
  trig, 19
\\tkz Axe XY, 21
\\tkz Axe XY[\{local options\}], 21
\\tkz Axe X[\{local options\}], 19
\\tkz Axe Y, 21
\\tkz Axe Y[\{local options\}], 21
\\tkzClip[space], 46
\\tkzClip, 7, 45, 46
\\tkz Clip: options
  space, 45
\\tkzClip BB, 7, 43, 44
\\tkzClipOutPolygon, 47
\\tkz Clip[\{local options\}], 45
\\tkzDefPoint, 29, 30, 34, 55, 62
\\tkzDefPoint: arguments
  a:r, 29
  x,y, 29
\\tkzDefPoint: options
  shift, 29
\\tkzDefPoints(0/0/0/2/4/\text{A}), 31
\\tkzDefPoints, 31, 40, 55
\\tkzDefPoints: arguments
  x_i/y_i, 31
\\tkzDefPoints[\{local options\}]{(x_i/y_i/n_1, x_i/y_i/n_2, …)}, 31
\\tkzDefPoint[\{local options\}]{(x,y)}{(name)} ou \((a:r)\){(name)}, 29
\\tkzDefRandPointOn, 7
\\tkzDefSetOfPoints, 40, 55
\\tkzDefSetOfPoints: arguments
  x_i/y_i, 40
\\tkzDefSetOfPoints: options
  prefix, 40
\\tkzDefSetOfPoints[\{local options\}]{x_i/y_i/x_i/y_i, … , x_i/y_i}, 40
\\tkzDefShiftPoint, 31
\\tkzDefShiftPoint: arguments
  a:r, 31
  (x,y), 31
\\tkzDefShiftPointCoord, 32
\\tkzDefShiftPointCoord: arguments
  a:r, 32
  (x,y), 32
\\tkzDefShiftPointCoord: options
  a,b, 32
\\tkzDefShiftPointCoord[(a,b)]{(x,y)}{(name)} ou \( (a:r)\){(name)}, 32
\\tkzDefShiftPoint[(Point)]{(x,y)}{(name)} ou \( (a:r)\){(name)}, 31
\\tkzDrawMark, 55, 57
\\tkzDraw Mark: options
  prefix, 57
\\tkzDraw Marks, 57, 58
\\tkzDrawMarks, 57, 58
\\tkzDrawMarks: options
  prefix, 57
\\tkzDraw Marks[\{local options\}]{(\textit{list} of points)}, 57
\\tkzDraw Mark[\{local options\}]{(\textit{point})}, 57
\\tkzDraw Point(A,B), 62
\\tkzDraw Point, 33, 62
\\tkzDrawPoint, 33
\\tkzDraw Point: arguments
  point, 33
\\tkzDraw Point: options
  color, 33
  shape, 33
  size, 33
\\tkzDrawPoints(A,B,C), 34
\\tkzDrawPoints, 34, 62
\\tkzDrawPoints: arguments
  points list, 34
\\tkzDrawPoints[\{local options\}]{(\textit{list} of points)}, 34
\\tkzDraw Point[\{local options\}]{(\textit{point})}, 33
\\tkzDraw Segment, 7
\\tkzDraw Segments(B,B’ C,C’), 62
\\tkzDraw SetOfPoints, 55
\\tkzDraw SetOfPoints: options
  prefix, 55
\\tkzDraw SetOfPoints[\{local options\}], 55
\\tkzDraw X, 15, 19, 21, 22
\tkzPointShowCoord: options
 noxdraw, 39
  xlabel, 39
  xstyle, 39
\tkzPointShowCoord[(local options)](point), 39
\tkzRep: options
  colorlabel, 49
  color, 49
  line width, 49
  posxlabel, 49
  posylabel, 49
  xlabel, 49
  xnorm, 49
  ylabel, 49
  ynorm, 49
\tkzRep[(local options)], 49
\tkzSaveBB, 44
\tkzSaveBB(⟨A⟩), 44
\tkzSetBB(⟨x_A; y_A⟩), 44
\tkzSetBB(⟨x_B; y_B⟩) ou bien (⟨A⟩) (⟨B⟩), 44
\tkzSetOfPoints, 40, 55
\tkzSetUpAxis, 24
\tkzSetUpAxis[local options], 23
\tkzSetUpAxis: options
  font, 23
  line width, 23
  ticka, 23
  tickb, 23
  tickwd, 23
\tkzSetUpAxis[(local options)], 23
\tkzSetUpMark[mark=heart], 56
\tkzSetUpMark, 56
\tkzSetUpMark[local options], 56
\tkzsetUpPoint, 37, 38
\tkzSetUpPoint: options
  color, 37
  fill, 37
  shape, 37
  size, 37
\tkzSetUpPoint[(local options)], 37
\tkzShowBB, 43, 44
\tkzShowBB[(local options)], 43
\tkzText, 59
\tkzText: options
  color, 59
  fill, 59
  opacity, 59
  text, 59
\tkzText[(local options)]{(dot)}{(text)}, 59
\tkzVLine{1}, 51
\tkzVLine, 51
\tkzVLine[local options], 51
\tkzVLine: arguments
  decimal number, 51
\tkzVLine[local options], 51
\tkzVLine: options
  color, 51
  line width, 51
  style, 51
\tkzVLine{1,4}, 52
\tkzVLine, 52
\tkzVLine[local options]{(list of values), 52
\tkzVLine[(local options)]{(decimal number), 51
\tkzVTick{1}, 53
\tkzVTick, 53
\tkzVTick: arguments
  decimal number, 53
\tkzVTicks{1,3}, 54
\tkzVTicks, 53, 54
\tkzVTicks: arguments
  decimal number, 54
\tkzVTicks[(local options)]{(decimal number), 54
\tkzVTicks[(local options)]{(decimal number), 53
\useasboundingbox, 43
\usepackage{english}{babel}, 13
\usetikzlibrary{babel}, 9
\usetikzlibrary{babel}, 9