Drawing histogram bars inside the LaTeX picture–environment

Rainer Schöpf
Institut für Physik
Johannes Gutenberg Universität

Abstract

This article describes an enhancement of the LaTeX picture–environment to draw histogram bars. It is written in the self documenting TeX format developed by Frank Mittelbach.

1 User interface

```
\typeout{^^JDocument style option 'histogr',
version 1.0 by RmS, released Nov 15, 1987}
```

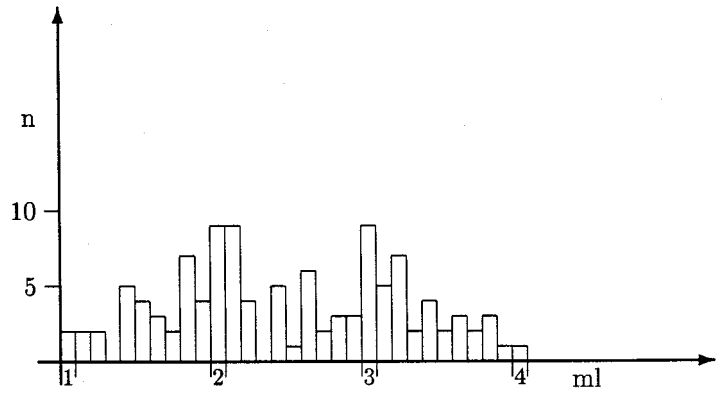
This is a macro collection to draw histogram bars inside a picture–environment. Use is as follows:

\histogram$(x_0,y_0)(x_1,y_1)...(x_n,y_n)$

\noverticallines
\verticallines

The coordinate pairs specify the upper left corner of the histogram bars, i.e. this will draw a horizontal line from (x_i, y_i) to (x_{i+1}, y_i), then a vertical line from (x_{i+1}, y_i) to (x_{i+1}, y_{i+1}) if \noverticallines was specified, else from (x_{i+1}, y_0) to $(x_{i+1}, \max(y_i, y_{i+1}))$. Default is \verticallines. y_0 should be less or equal the minimum of all the y_i (i.e. other cases have not been tested).

Let's start with an example: to get the following picture:



Behandler 1

I used these LaTeX commands:

```
\setlength{\unitlength}{1mm}
\begin {picture}(100,65)(-10,-15)

\thicklines
\put(0,-3){\vector(0,1){50}}
\put(-3,0){\vector(1,0){90}}
\thinlines

\put(0,0){\line(0,-1){2}}
\put(2,0){\line(0,-1){2}}
\put(20,0){\line(0,-1){2}}
\put(22,0){\line(0,-1){2}}
\put(40,0){\line(0,-1){2}}
\put(42,0){\line(0,-1){2}}
```

```
\put(60,0){\line(0,-1){2}}
\put(62,0){\line(0,-1){2}}

\put(0,-1){\makebox(2,0)[t]{\small 1}}
\put(20,-1){\makebox(2,0)[t]{\small 2}}
\put(40,-1){\makebox(2,0)[t]{\small 3}}
\put(60,-1){\makebox(2,0)[t]{\small 4}}
\put(70,-1){\makebox(0,0)[t]{ml}}

\put(0,10){\line(-1,0){2}}
\put(0,20){\line(-1,0){2}}

\put(-3,8){\makebox(0,4)[r]{5}}
\put(-3,18){\makebox(0,4)[r]{10}}
\put(-3,30){\makebox(0,4)[r]{n}}

\put(15,-10){Behandler 1}

\histogram(0,0)(0,4)(2,4)(4,4)(6,0)(8,10)(10,8)(12,6)(14,4)
(16,14)(18,8)(20,18)(22,18)(24,8)(26,0)(28,10)(30,2)
(32,12)(34,4)(36,6)(38,6)(40,18)(42,10)(44,14)(46,4)
(48,8)(50,4)(52,6)(54,4)(56,6)(58,2)(60,2)(62,0)
\end{picture}
```

2 Implementation

`\hist@x` `\hist@y` `\hist@ystart` Here's how it is implemented: first we allocate three counters that are needed later on. `\hist@x` and `\hist@y` are the x and y coordinate of the *current point*, i.e. the point that serves as a start for the next box of the histogram. `\hist@ystart` holds the y coordinate of the first point, i.e. y_0.

```
\newcount\hist@x
\newcount\hist@y
\newcount\hist@ystart
```

`\noverticallines` `\verticallines` We need a switch to decide if the vertical lines of the histogram boxes are to be drawn from y_i to y_{i+1} or from y_0 to $\max(y_i, y_{i+1})$. Default is the latter.

```
\newif\ifhist@vert

\let\verticallines\hist@verttrue
\let\noverticallines\hist@vertfalse

\hist@verttrue
```

`\histogram` The `\histogram` command takes the starting point as argument and initializes the counters. `\hist@x`, `\hist@y` and `\hist@ystart` are set to x_0, y_0 and y_0, respectively.

```
\def\histogram(#1,#2){\hist@x #1 \hist@y #2 \hist@ystart\hist@y
```

Then the macro `\hist@next` is used.

```
  \hist@next}
```

`\hist@next` `\hist@next` looks at the next token to see if there is another open parenthesis. If this is the case it calls `\hist@box`, otherwise `\hist@end`.

```
\def\hist@next{\@ifnextchar ({\hist@box}{\hist@end}}
```

`\hist@box` The macro `\hist@box` does nearly all the work. The first thing to do is to set the temporary counter `\@tempcnta` to $x_{i+1} - x_i$. Remember that `\hist@x` is the x coordinate of the last point (i.e. x_i) whereas the macro's first argument is x_{i+1}. So we write

```
\def\hist@box(#1,#2){\@tempcnta -\hist@x
  \advance\@tempcnta #1
```

The next step is easy: draw the horizontal part of the histogram box. The line starts at (x_i, y_i) and has length \@tempcnta\unitlength.

```
\ifnum \@tempcnta >\z@
     \put(\hist@x,\hist@y){\line(1,0){\@tempcnta}}\else
     \put(\hist@x,\hist@y){\line(-1,0){-\@tempcnta}}\fi
```

Now set \hist@x to x_{i+1}:

```
\hist@x #1
```

If \verticallines was set we first set \@tempcnta to $\max(y_i, y_{i+1})$:

```
\ifhist@vert
   \ifnum \hist@y >#2 \@tempcnta\hist@y
     \else \@tempcnta #2 \fi
```

then we set \@tempcntb to the same value and \@tempcnta to the length of the line to draw.

```
\@tempcntb\@tempcnta
\advance\@tempcnta -\hist@ystart
```

We draw the line

```
\put(\hist@x,\@tempcntb){\line(0,-1){\@tempcnta}}%
```

which finishes this case.

```
\else
```

In the other case (i.e. if \noverticallines was set) we have to draw a line from y_i to y_{i+1}. We set \@tempcnta to $y_{i+1} - y_i$

```
\@tempcnta -\hist@y
\advance\@tempcnta #2
```

and draw the line.

```
\ifnum \@tempcnta >\z@
   \put(\hist@x,\hist@y){\line(0,1){\@tempcnta}}\else
   \put(\hist@x,\hist@y){\line(0,-1){-\@tempcnta}}\fi
```

Thus endeth the drawing.

```
\fi
```

Finally we set \hist@y to y_{i+1} and call \hist@next to look for the next coordinate pair.

```
\hist@y #2\hist@next}
```

\hist@end There is only one thing we left out: what if there is not another open parenthesis? That's the easy part: do nothing.

```
\def\hist@end{}
```

Frank Mittelbach has suggested that the x–coordinate should specify the midpoint of the histogram bar, not the upper left corner. However, I don't see how this will work if the bars have different widths. What do you think about it? Well, that's all. Use it and enjoy.