The bardiag package *

R. Stepanyan
rstepanyan@yahoo.com

December 10, 2003

Abstract

The bardiag package is inspired and entirely based on PSTricks package. The main purpose is to make the drawing of (simple) bar diagrams possible and (I hope) easy in LaTeX.

1 Package installation

The bardiag package consists of two sty-files: `bardiag.sty` (written by me) and `pstfp.sty` which is an almost exact copy of the `fp-basic.sty` package by Michael Mehlich. The second one is needed to work with floating point numbers and has been patched to work with `multido.sty` from the PSTricks . The `bardiag.sty` also loads `bardiag.bar` with some internal routines.

The bardiag depends on `pstricks`, `pstcol`, `pst-grad`, `multido`, `calc`, `fp-snap`, `ifthen`, so make sure they are installed. Then copy the `bardiag.sty`, `bardiag.bar`, and `pstfp.sty` files into your texmf tree, where LaTeX is able to find them. That's it!

This manual consists of three files: this one (`bardiag.ps`), `bardiag1.ps`, and `bardiag2.ps`. The present one describes the installation and features needed to draw "vertically" oriented diagrams, i.e. with bars parallel to the Y-axis; `bardiag1.ps` explains how to build "horizontal" bar diagrams; and `bardiag2.ps` includes all appendices, including some source listings and examples of alternative shapes, not included in the bardiag style-file itself, but available as a config file.

In the `example` directory you can find: (i) `diagrams.tex` which contains some examples of colored diagrams; (ii) `diagramsbw.tex` with b/w diagrams – it is shown how to use pattern filling instead of color; (iii) `altdiags.tex` represents some "alternative" bar shapes.

2 Quick start

2.1 Simple bar diagram

Let me start from an example. One wants to draw a diagram, representing table 1. It can be

Year	1998	1999	2000	2001	2002
Income	120	123	147	176	132

Table 1: See diagram Fig.2.1.

done with the following piece of code

*This file has version number 0.5 and describes bardiag ver.0.5

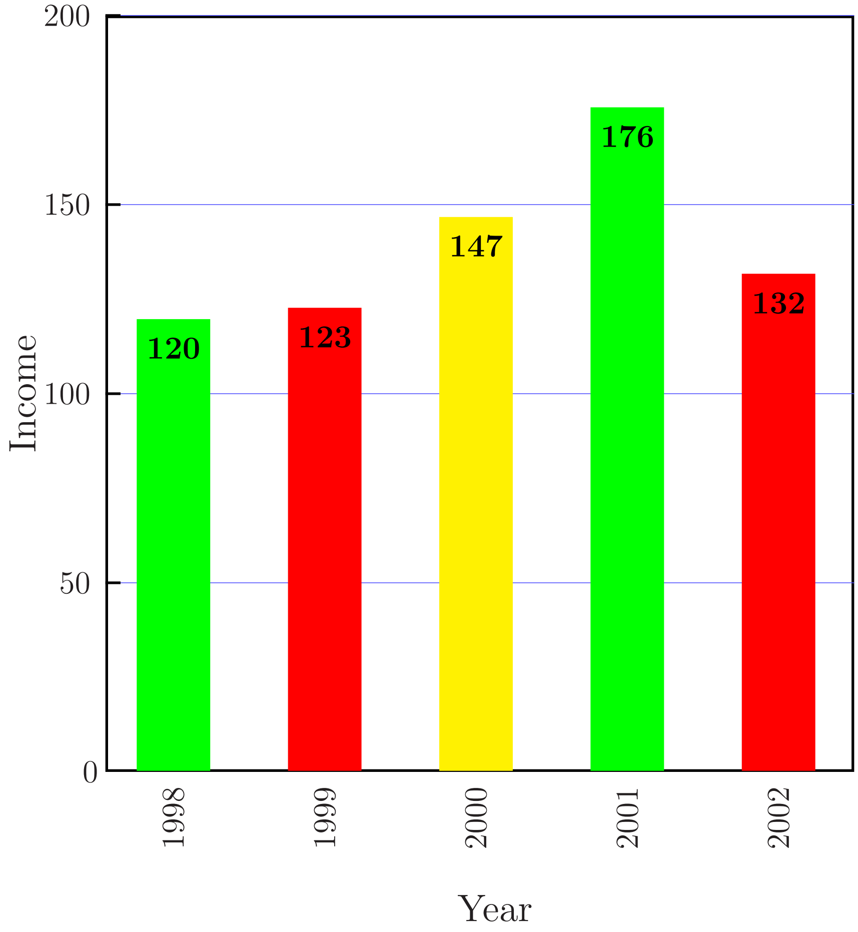


Figure 2.1: Example 1 to table 1.

```
\renewcommand{\betweenticks}{50}
\bardiagrambegin{9.5}{200}{2cm}{1}{2}{1cm}{0.05cm}
    \drawlevellines
    \baritem{1998}{120}{green}
    \baritem{1999}{123}{red}
    \baritem{2000}{147}{yellow}
    \baritem{2001}{176}{green}
    \baritem{2002}{132}{red}
\bardiagramend{{\large Year}}{{\large Income}}
```

The result is Fig.2.1. So, a bit of explanation now.

Any diagram begins with `\bardiagrambegin` and ends with `\bardiagramend` commands. Arguments of the first one are

`\bardiagrambegin{diagheight}{diagwidth}{bottomheight}{dbar}{dx}{xunit}{yunit}`

The meaning of the first 5 arguments is apparent from the Fig.2.2. The 6th and the 7th parameters are unit lengths along x- and y-axis correspondingly. Note, that `diagheight`, `diagwidth`, `dbar`, and `dx` are NOT lengths, but (floating point) numbers. They are then converted to lengths automatically using `xunit` and `yunit`. However `bottomheight` is a "real" length and should be accompanied, exactly as `xunit` and `yunit`, by `cm` or `in` or whatever you find more comfortable for yourself. This allows to scale the diagram changing `xunit` and/or `yunit` but leave the bottom part of the same height. Just to demonstrate that I drew the same diagram as Fig.2.1, but used a smaller last argument in `bardiagrambegin`:

`\bardiagrambegin{9.5}{200}{2cm}{1}{2}{1cm}{0.01cm}`

As you see on Fig.2.3 the bottom remains readable.

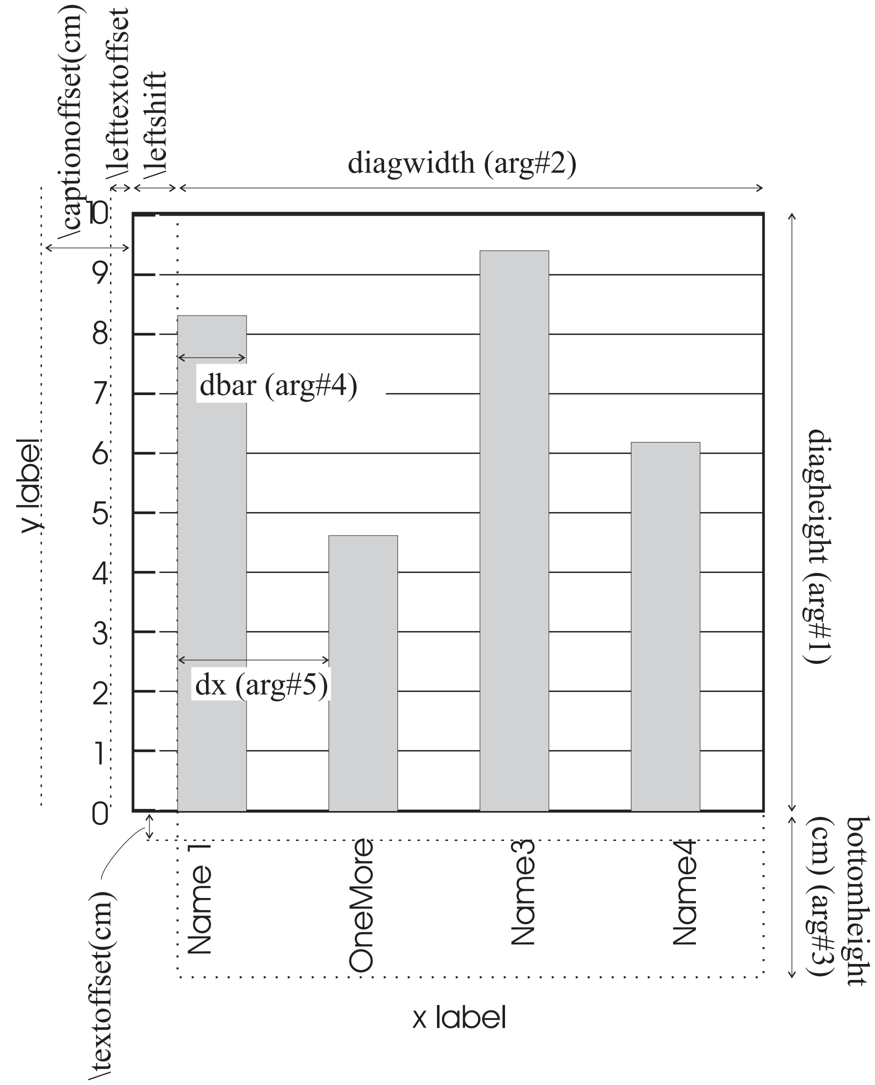


Figure 2.2: Arguments (shown with #1..5) of the `bardiagrambegin` command and other parameters

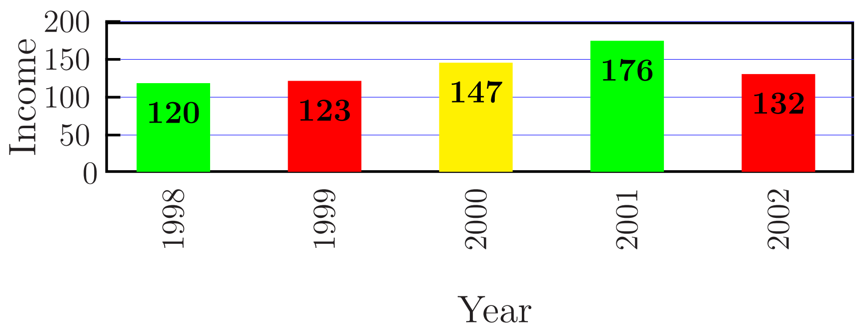


Figure 2.3: The same data as in Fig.2.1 but `yunit` is 0.1cm instead of 0.5cm.

The `\bardiagrambegin` begins a bar diagram and `\bardigramend` ends it. It has two arguments: x and y label. In the example above they are `{\large Year}` and `{\large Income}` correspondingly.

Between the "begin" and "end" statements one actually "draws the stuff". Command `\baritem{caption}{value}{color}` Draws a bar of height `value` using color `color` and prints `caption` below it. By default it also prints numbers (values) near the top of the bar - how to change this behavior is explained further on [section 3.3].

Command \drawlevellines draws horizontal lines and put ticks with numbers along the y-axis. An important parameter in this case is the \betweenticks . It determines the distance between ticks (level lines) in units of yunit. Note that the \drawlevellines can be issued at any point inside the diagram. In this way bars can lie over (as in Fig.2.1) or be behind the level lines.

2.2 Adding a legend

The next step is to add a legend to the diagram:

```
\renewcommand{\betweenticks}{50}
\bardiagrambegin{9.5}{200}{2cm}{1}{2}{1cm}{0.05cm}
    \drawlevellines
    \baritem{1998}{120}{green}
    \baritem{1999}{123}{red}
    \baritem{2000}{147}{yellow}
    \baritem{2001}{176}{green}
    \baritem{2002}{132}{red}
    \diagLegendbegin{0.1}{203}{5}
      \diagLegenditem{Good statistics}{green}
      \diagLegenditem{Moderate statistics}{yellow}
      \diagLegenditem{Bad statistics}{red}
    \diagLegendend
\bardiagramend{{\large Year}}{{\large Income}}
```

The result is shown in Fig.2.4.

Any legend begins with \diagLegendbegin{Xupleft}{Yupleft}{width}, where the first two arguments represent the upper left point of the legend's bounding box (see Fig.2.5) and the last one its width. The height of the box is calculated automatically.

\diagLegenditem{caption}{icolor} produces a rectangle of the color icolor and prints caption next to it. To finish the legend the \diagLegendend is used. Note, that *all* these coordinates are in xunit and yunit units, not in cm or in!

Some properties of the legend can be changed using
\diagLegendoptions{framefillcolor}{framelinecolor}{framelinewidth} command.

2.3 Groups of items

I can imagine that you need more than just bars with figures. Let me introduce one more feature: grouping of items as in Fig.2.6. This grouping is done using
\subbaritem{caption}{value}{color}.
So, a piece of code like

```
\baritem{1998}{123}{red}
    \subbaritem{}{118}{blue}
```

will produce *two* bars next to each other. You can go further and make groups of 3, 4, etc bars. In this way

```
\baritem{1998}{123}{red}
    \subbaritem{}{118}{blue}
    \subbaritem{}{108}{green}
```

would produce group of 3 bars.

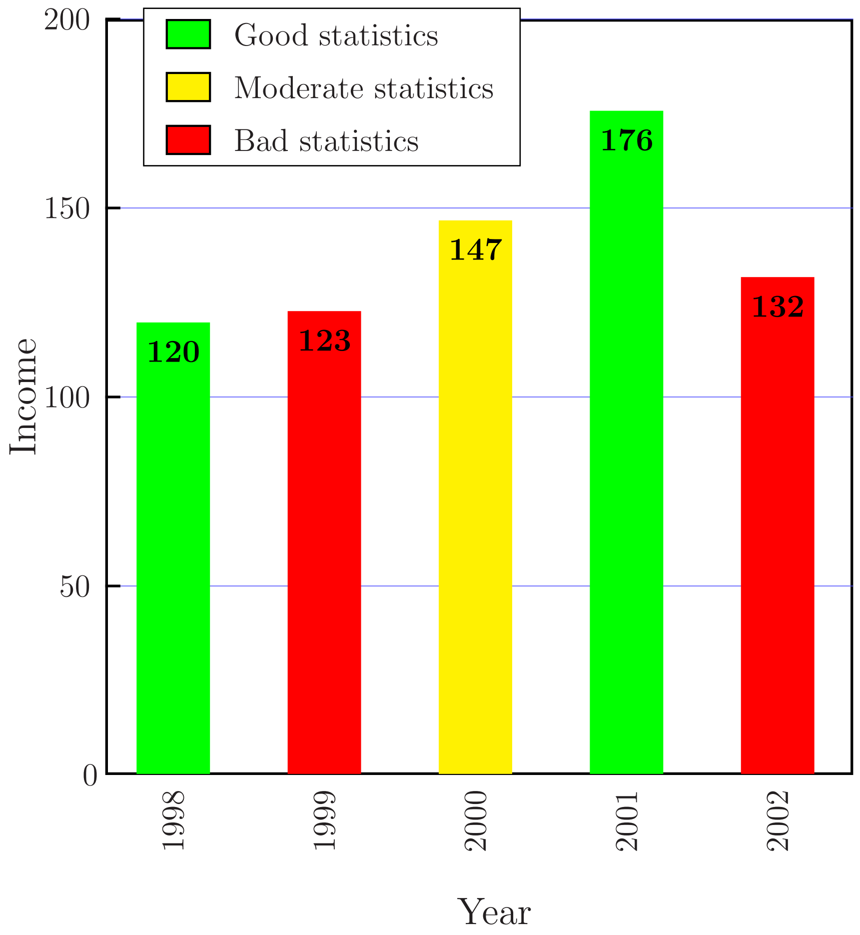


Figure 2.4: Adding legend to Fig.2.1

The “bad news” about Fig.2.6 is that the labels (year) are not centered anymore. To correct the situation, just issue the command `\barspergroup{2}` just after the beginning of the chart. It commands the BagDiag to place the label assuming each group of items to be 2 bars wide.

Another way of grouping items is put them on top of each other as in Fig.2.7, where, actually, two these approaches are combined. The above explained `\barspergroup` trick is used here too. The corresponding source code is shown below:

```
\renewcommand{\betweenticks}{50}
\bardiagrambegin{12.5}{210}{2cm}{1}{2.5}{0.8cm}{0.04cm}
    \barspergroup{2}
    \drawlevellines
    \baritem{1998}{23}{lightred}
      \subtopbaritem{}{100}{red}
    \subbaritem{}{28}{lightblue}
      \subtopbaritem{}{90}{blue}
    %
    \baritem{1999}{37}{lightred}
      \subtopbaritem{}{89}{red}
    \subbaritem{}{38}{lightblue}
      \subtopbaritem{}{81}{blue}
    %
    \baritem{2000}{43}{lightred}
      \subtopbaritem{}{101}{red}
    \subbaritem{}{52}{lightblue}
```

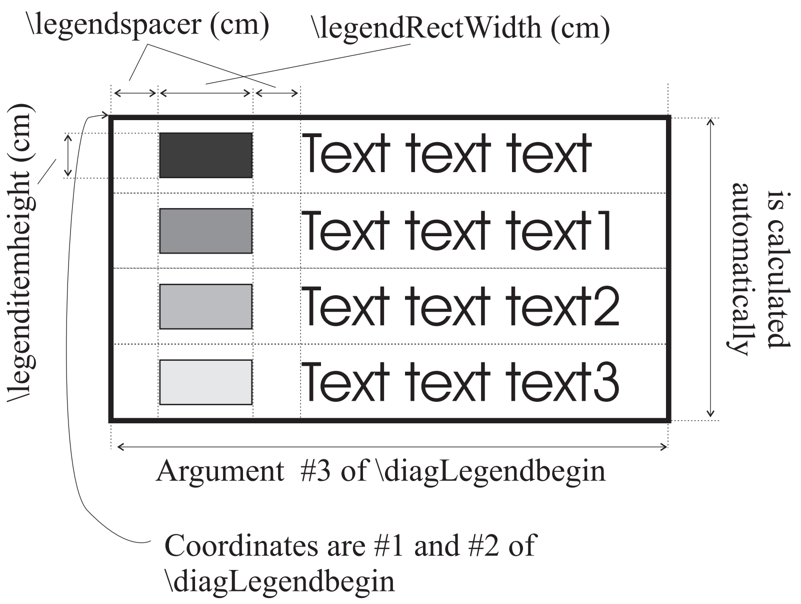


Figure 2.5: Legend-related parameters

```
      \subtopbaritem{}{96}{blue}
    %
    \baritem{2001}{83}{lightred}
      \subtopbaritem{}{93}{red}
    \subbaritem{}{92}{lightblue}
      \subtopbaritem{}{78}{blue}
    %
    \baritem{2002}{64}{lightred}
      \subtopbaritem{}{64}{red}
    \subbaritem{}{70}{lightblue}
      \subtopbaritem{}{64}{blue}
    %
    \diagLegendbegin{0}{222}{7.3}
      \diagLegenditem{\it Women older than 30}{red}
      \diagLegenditem{\it Women younger than 30}{lightred}
      \diagLegenditem{\it Men older than 30}{blue}
      \diagLegenditem{\it Men younger than 30}{lightblue}
    \diagLegendend
\bardiagramend{{\large Year}}{{\large Income}}
```

The command `\subtopbaritem` has the same arguments as other "baritem" commands and produces a bar on top of the previous "baritem", which can be `\baritem`, `\subbaritem` or even `\subtopbaritem`.

2.4 Making a 3D frame

Frame can be done 3-dimensional by using the switch `\frameTD`. Its default value is 0, however, using
`\renewcommand{\frameTD}{1}`
a 3D frame can be forced. To control the 3D effect, parameters `\tdx` and `\tdy` are used [section 3.5]. Here, Fig.8(a), is an example of the same diagram as in Fig.2.1 but with a 3D frame

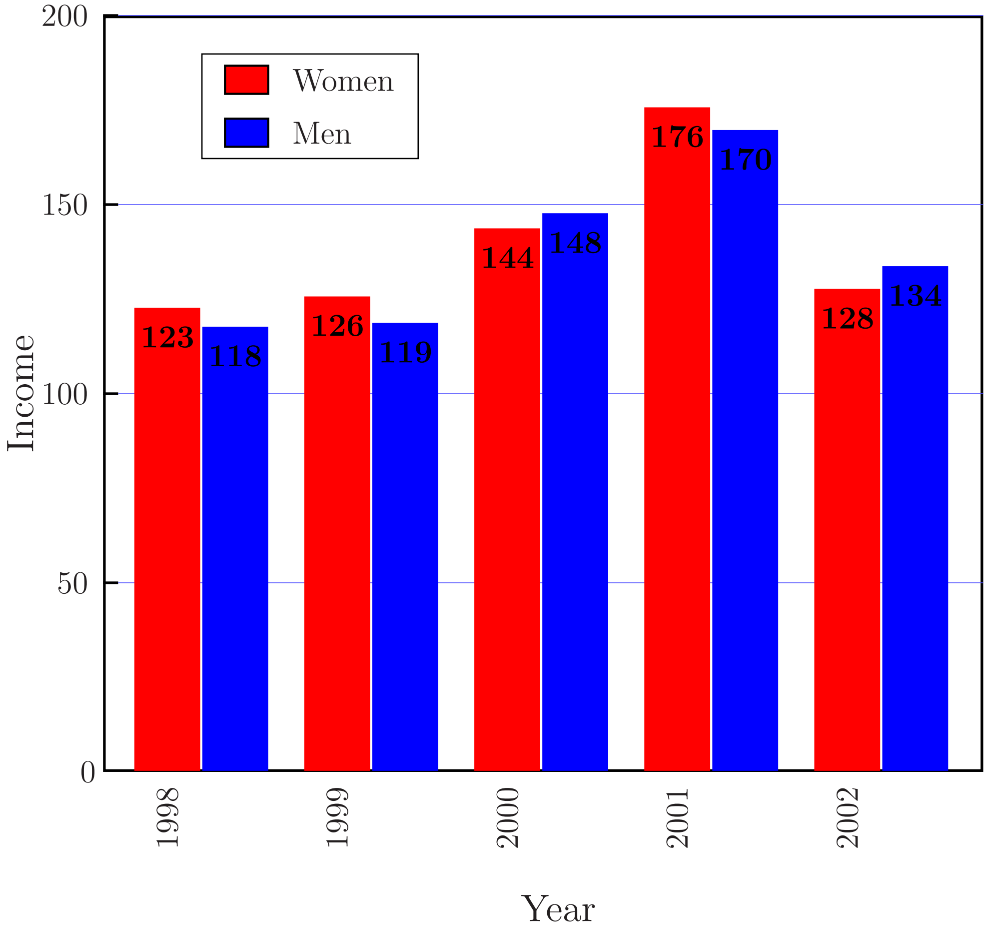


Figure 2.6: Grouped items

2.5 Adding shadow to the legend

You can force the legend to have a shadow by
`\newcommand{\legendShadow}{1}`.
However, if you use a 3D frame this will be done automatically.

2.6 Changing the item-bar shape

The `bardiag` package provides also 3D item-bar shapes: rectangular 3d and cylindrical. In fact, all the "bar-item" commands, such as `\XXXbaritem`, call the `\ActiveBarPrimitive` command to draw "a bar". This means that you can provide your own shapes [section 3.3].

Concerning the shapes, provided by `bardiag` : Fig.8(b) is an example of `\renewcommand{\ActiveBarPrimitive}{\barTDrect}` command. The default value , `\barRect`, produces a flat rectangle and `\barCylinder` a cylinder. All these shapes can be grouped, etc, exactly in the same way as flat rectangular bars.

The package provides: `\barRect`, `\barTDRect`,`\barCylinder`. Some additional shapes are discussed in `bardiag2.ps`.

2.7 Adding error margins ("whiskers")

Adding "whisker" is simple! Just issue `\barerrormargins{dYdown}{dYup}` command after the corresponding `\baritem` or `\subbaritem`[1].

[1]Unfortunately, it will not work with `subtopbaritem`.

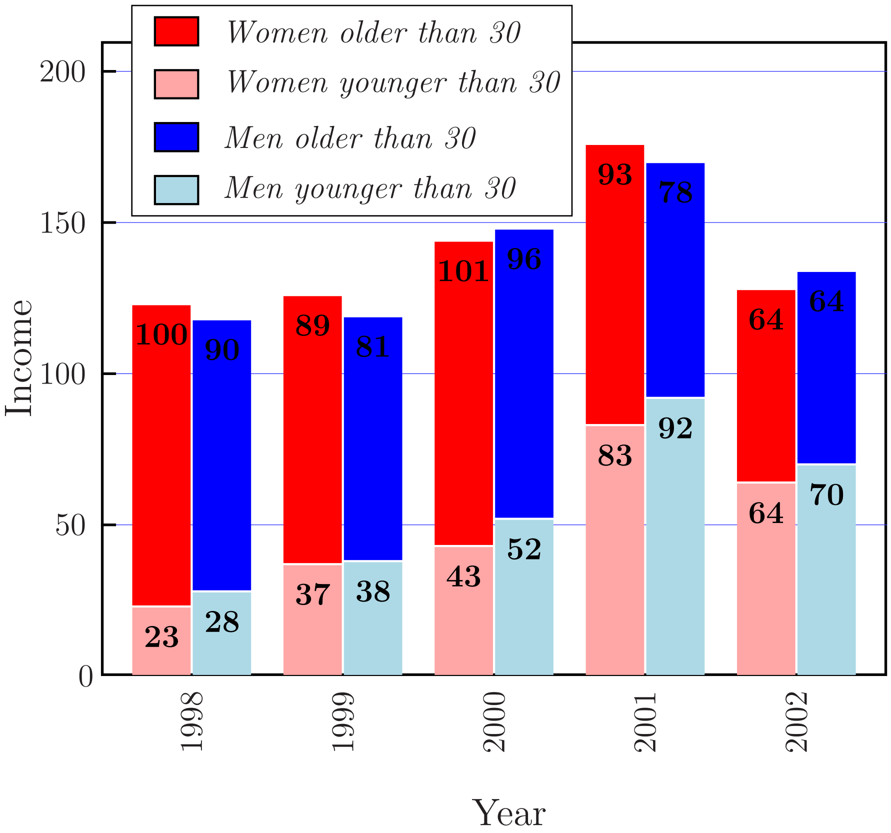
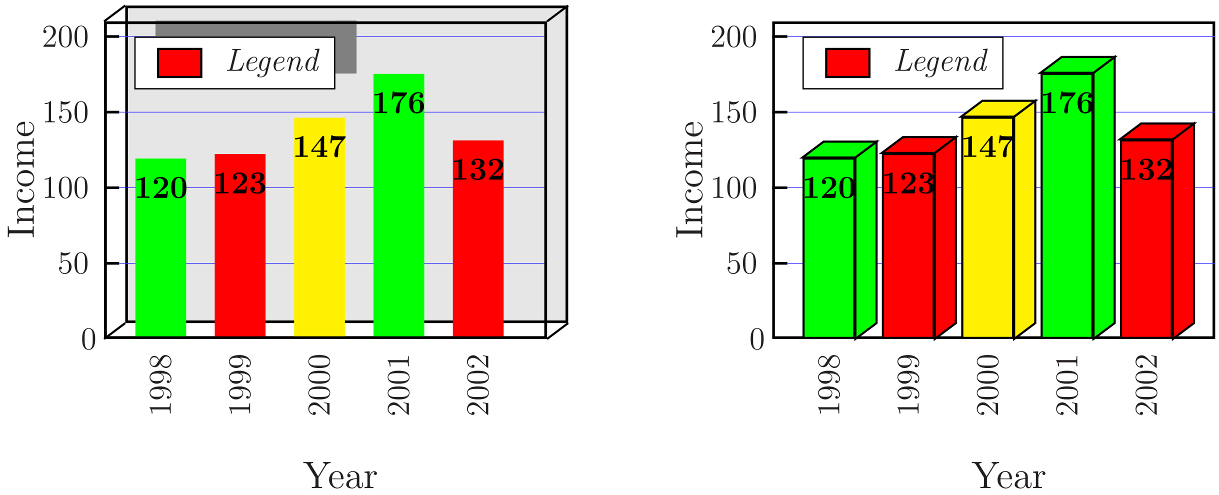


Figure 2.7: Grouped items



(a) 3D frame

(b) 3D items

Figure 2.8: 3D effects

```
\baritem{1998}{123}{gray} \barerror{25}
\subbaritem{}{118}{lightgray} \barerrormargins{35}{15}
```

If you need symmetric whisker, use `\barerror{dY}` instead. The properties of the "whisker"-line are controlled by two parameters:
(i) `\barerrorterm` which defines the terminators. Default is `\newcommand{\barerrorterm}{*-|}`, as in Fig.2.9
(ii) `barerrorstyle` can change the color, thickness, etc; use `\newpsstyle` to redefine it. Default: `\newpsstyle{barerrorstyle}{linewidth=2pt,linecolor=black}`.

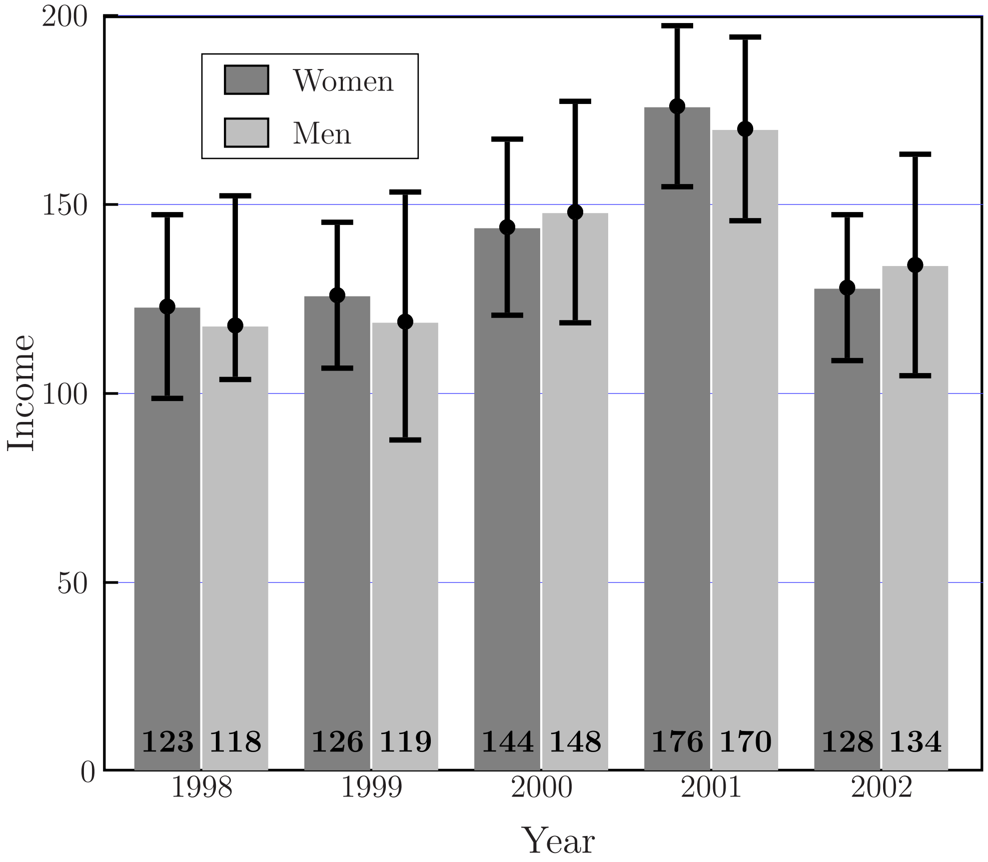


Figure 2.9: Grouped items and “whiskers”

Everything said above holds for “horizontal” diagrams (`bardiag1.ps`) as well.

2.8 Pattern fillings and black-and-white documents

In many cases one cannot use any colors, except grayscale: not so many publications are printed in color. So, it would be handy to use patterns to fill the bar-items. This is possible, because `\baritem` and friends take also an optional argument: style. In fact, the default one is `fillstyle=solid` to get solid filled rectangles. In fact, one can call

`\baritem[fillstyle=vlines*]{xxx}{nnn}{gray}`

to produce a dashed bar. To get more info about fill styles, look into the `pstricks` documentation: e.g. `psr-usr2.ps`, pp. 27–28. Three of the relevant styles are: `vlines*`, `hlines*`, and `crosshatch*`. Also parameters like `hatchwidth`, `hatchsep`, `hatchcolor`, `hatchangle` can be used. Both

`\baritem[fillstyle=vlines*,hatchsep=8pt,hatchangle=30]{xxx}{nnn}{gray}`

and

```
\newpsstyle{myfill}{fillstyle=vlines*,hatchsep=8pt,hatchangle=30}
\baritem[style=myfill]{xxx}{nnn}{gray}
```

are acceptable and produce the same. A comprehensive example to study is presented in Fig.2.10:

```
\renewcommand{\betweenticks}{100}
\renewcommand{\shownumbers}{0}

\definecolor{gA}{gray}{0.5}
\definecolor{gB}{gray}{0.7}
\definecolor{gC}{gray}{0.9}
```

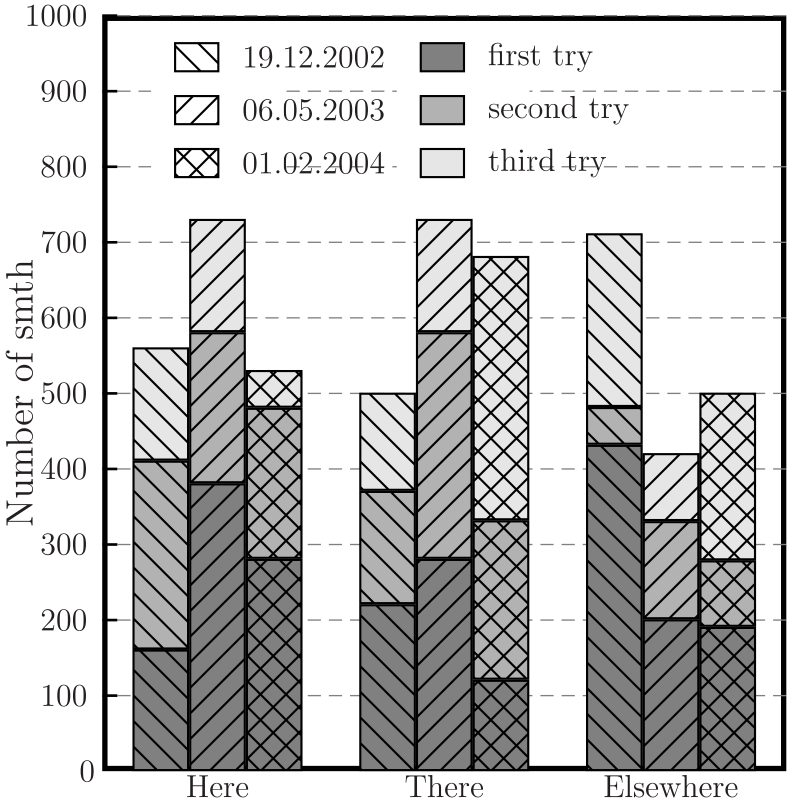


Figure 2.10: Using optional style argument of “baritem” and “diagLegenditem”

```
\newpsstyle{diagframestyle}{linewidth=2pt,linecolor=black}
\newpsstyle{stA}{linestyle=solid,hatchsep=6pt,fillstyle=vlines*}
\newpsstyle{stB}{linestyle=solid,hatchsep=6pt,fillstyle=hlines*}
\newpsstyle{stC}{linestyle=solid,hatchsep=6pt,fillstyle=crosshatch*}

\bardiagrambegin{17.3}{1000}{0.5cm}{1.5}{6}{0.5cm}{0.01cm}
       \newpsstyle{levellinestyle}{linewidth=0.5pt,linecolor=gray,linestyle=dashed}
       \renewcommand{\barlabelangle}{0}

       \barspergroup{3}

       \drawlevellines
       \baritem[style=stA]{Here}{161}{gA}
         \subtopbaritem[style=stA]{}{250}{gB}
         \subtopbaritem[style=stA]{}{150}{gC}
        \subbaritem[style=stB]{}{381}{gA}
         \subtopbaritem[style=stB]{}{200}{gB}
         \subtopbaritem[style=stB]{}{150}{gC}
        \subbaritem[style=stC]{}{281}{gA}
         \subtopbaritem[style=stC]{}{200}{gB}
         \subtopbaritem[style=stC]{}{50}{gC}
       %
       \baritem[style=stA]{There}{221}{gA}
         \subtopbaritem[style=stA]{}{150}{gB}
         \subtopbaritem[style=stA]{}{130}{gC}
        \subbaritem[style=stB]{}{281}{gA}
         \subtopbaritem[style=stB]{}{300}{gB}
         \subtopbaritem[style=stB]{}{150}{gC}
```

```
        \subbaritem[style=stC]{}{121}{gA}
         \subtopbaritem[style=stC]{}{211}{gB}
         \subtopbaritem[style=stC]{}{350}{gC}
       %
       \baritem[style=stA]{Elsewhere}{432}{gA}
         \subtopbaritem[style=stA]{}{50}{gB}
         \subtopbaritem[style=stA]{}{230}{gC}
        \subbaritem[style=stB]{}{201}{gA}
         \subtopbaritem[style=stB]{}{130}{gB}
         \subtopbaritem[style=stB]{}{90}{gC}
        \subbaritem[style=stC]{}{191}{gA}
         \subtopbaritem[style=stC]{}{88}{gB}
         \subtopbaritem[style=stC]{}{222}{gC}
       %
       % we can call
       \diagLegendoptions{white}{white}{0pt}
       % to produce a legend without a frame
%
       \diagLegendbegin{0.5}{980}{3.5}
         \diagLegenditem[style=stA]{19.12.2002}{white}
         \diagLegenditem[style=stB]{06.05.2003}{white}
         \diagLegenditem[style=stC]{01.02.2004}{white}
       \diagLegendend
       %
       \diagLegendbegin{7}{980}{3.5}
         \diagLegenditem{first try}{gA}
         \diagLegenditem{second try}{gB}
         \diagLegenditem{third try}{gC}
       \diagLegendend
\bardiagramend{}{\large Number of smth}
```

As shown in this example, the same technique applies to `\diagLegenditem`.

Note, not all "bar shapes" will make use of the style you supply: e.g., there is no use in pattern-filled 3D cylinders because they look flat. All the bar-primitives do accept the style option. It should always work well for `\barRect` and `\barArrowRect`. In the case of `\barTDRect` you can try to play around – personally, I do not see much use of it. Such primitives as `\barCylinder`, `\barGradRect`, `\barGradMidRect`, `\barGradCylinder` will just ignore this optional argument.

2.9 Additional information

Try to have a look at the examples, which can be found in the `example` directory.

3 A more sophisticated example

See Fig.3.1. More examples can be found in `bardiag1.ps` and `bardiag2.ps`. In `bardiag1.ps` it is also explained how to draw a “horizontally” oriented diagram.

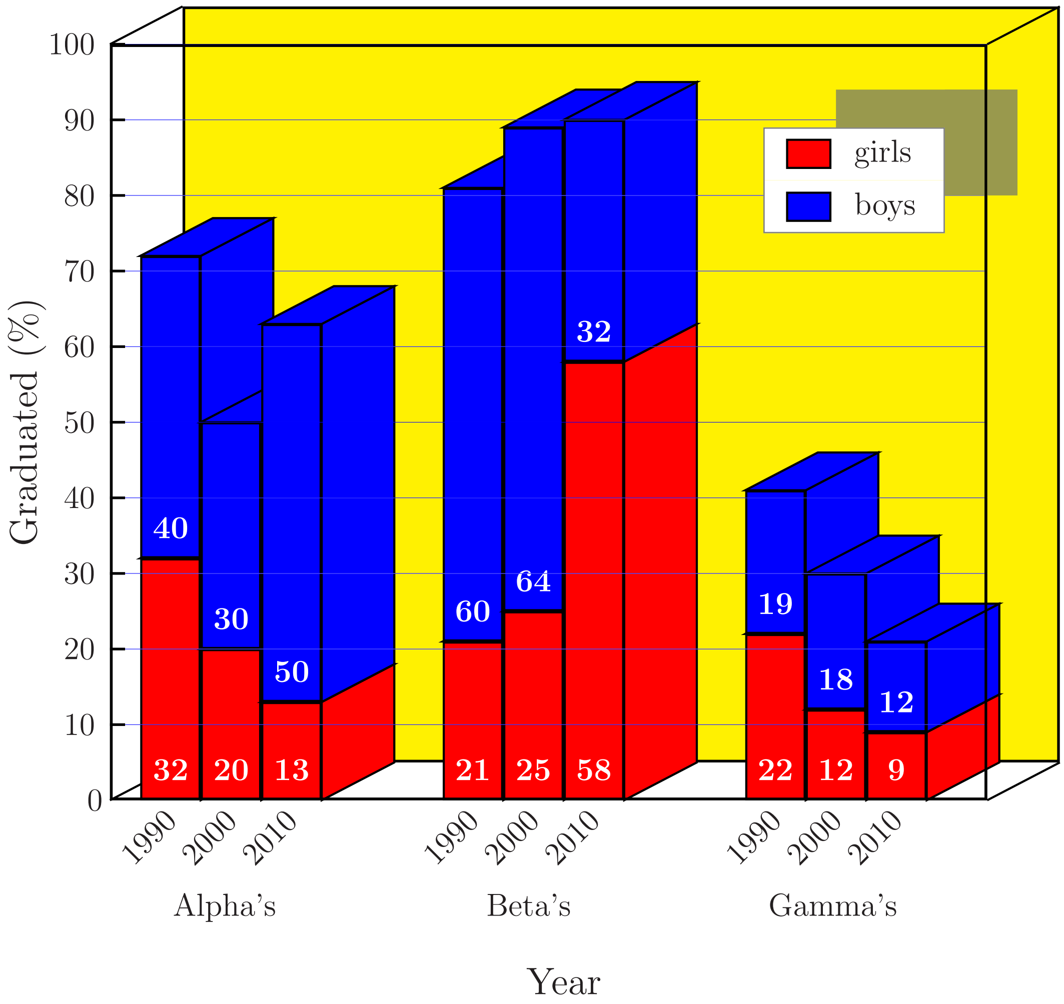


Figure 3.1: A more sophisticated example. See `bardiag2.ps` for the source code.

3.1 Frame properties

1. `\bardiagrambegin{#1}{#2}{#3}{#4}{#5}{#6}{#7}` see Fig.2.2

   #1 diagram height (number)

   #2 diagram width (number)

   #3 bottom height (length, use `cm`, `in`,. . .)

   #4 bar width (number)

   #5 distance between (groups) of bars

   #6 x unit length (`cm`, `in`,. . .)

   #7 y unit length (`cm`, `in`,. . .)

2. `\bardiagramend{#1}{#2}`

   #1 X label (text, things like `\bf`,`\it`, color can be used)

#2 Y label (the same)

3. Parameters to be changed using `\renewcommand`:
   - `\frameTD` (default 0) can 1 or 0; makes the frame 3D or 2D
4. Parameters to be changed using `\setlengt`:
   - `\leftshift` (default `\ticklength`+2mm) is the distance between the first bar and the frame left edge
   - `\textoffset` (default -0.2mm) see Fig.2.2
   - `\captionoffset` (default 1.5cm) see Fig.2.2
5. Styles, use `\newpsstyle` to change
   - `diagframestyle` is style of a 2D frame; defined as
     `\newpsstyle{diagframestyle}{linewidth=1pt,linecolor=black,fillstyle=none}`
   - `diagTDframestyle` is style of the foreground of a 3D frame;
     ```
     \newpsstyle{diagTDframestyle}
             {linewidth=1pt,linecolor=black,fillstyle=none}
     ```
   - `diagbgframestyle` is style of the background of a 3D frame;
     ```
     \newpsstyle{diagbgframestyle}
                 {linewidth=1pt,linecolor=black,fillcolor=frameBGgray}
     ```
     Also the definition `\definecolor{frameBGgray}{gray}{0.9}` is used.

3.2 Level lines and ticks

Both are produced by the `\drawlevellines` command. It can be issued at *any* point inside a diagram.

1. Parameters to be changed using `\renewcommand`:
   - `\betweenticks` is the distance between ticks/level lines in units of `yunit`, so no `cm` allowed
   - `\MAXnumberofticks` (default is 20) is the maximal number of ticks/level lines allowed. If `\betweenticks` is so small that the number of ticks exceeds `\MAXnumberofticks`, the latter will be used (`bardiag` will complain however, so a warning will be issued).
   - `\barlabelangle` determines the angle of labels to be written below the bars. It has to be changed *after* `\bardiagrambegin` is called, otherwise it will automatically be set to 90 degrees for vertical bars and 0 for horizontal ones.
   - `\ticklabelangle` is almost the same as `barlabelangle` but for the tick labels. It can be set at any point, not only after the beginnig of a diagram.
2. Parameters to be changed using `\setlengt`:
   - `\ticklength` (default 2mm) is length of a tick on the y-axis
   - `\lefttextoffset` (default `\leftshift`+2mm) see Fig.2.2
3. Styles, use `\newpsstyle` to change
   - `tickstyle` is the tick style; defined as
     `\newpsstyle{tickstyle}{linewidth=1pt,linecolor=black}`
   - `levellinestyle` is the level line style;
     `\newpsstyle{levellinestyle}{linewidth=0.2pt,linecolor=blue}`

3.3 Bars

1. `\baritem{name}{value}{color}` is used to show a "bar". Actually it calls `\ActiveBarPrimitive` to draw the "bar".

2. `\subbaritem{}{}{}` is used to produce a bar next to the previous bar (on the right hand side) without any space in between.

3. `\subtopbaritem{}{}{}` is used to produce a bar on top of the previous bar.

4. `\ActiveBarPrimitive` is used by all "baritem" commands to draw something that represents a bar. In fact, the `\ActiveBarPrimitive` is just a "pointer" to a command, which will really do the job. `bardiag` provides 3 such commands: `\barRect`, `\barTDRect`, and `\barCylinder`. Use, e.g., `\renewcommand{\ActiveBarPrimitive}{\barCylinder}` to draw cylinders instead of default rectangles. You can write your own "shape" (say `\mybar`) and put it somewhere in the preamble of the document or in the `bardiag.cfg` file and then point `\ActiveBarPrimitive` to it [see `bardiag.cfg` in `example` directory]. The `\mybar` should accept 5 arguments: (#1,#2) and (#3,#4) are coordinates of the bottom-left and upper-right corners and #5 is the color. In the `bardiag.cfg` file, which can be found, e.g., in the examples directory, a couple of "alternative" shapes are defined. Examples can be found in `altdiag.tex` and in the appendix (see file `bardiag2.tex` or `bardiag2.ps`).

5. Parameters to be changed using `\renewcommand`:

   - `\shownumbers` (default 1) use 1 to show and 0 to hide the numbers on item-bars
   - `\numbercolor` (default `\black\bf`) is the color of these numbers
   - `\placenumber` (default `\belowtop`) determines where on the bar put the number; can be `\belowtop`, `\overtop`, `\bottom`

6. Parameters to be changed using `\setlengt`:

   - `\numberoffset` (default `2\textoffset`) is offset of the number on a bar from the top (if `\overtop` or `\belowtop` are used) or the bottom (if `\bottom` is used)

3.4 Legend

1. `\diagLegendbegin{}{}{}` begins a legend. (#1,#2) is a coordinate of the top-left corner, #3 is the width of the bounding box (all in units, not is `cm` or `in`). It is assumed that diagram is started above. Each diagram can have any number of legends.

2. `\diagLegendend` ends a legend. It is needed to draw a shadow.

3. `\diagLegendoptions{}{}{}` is used to change the legend background color (#1), frame line color (#2), and frame line width (#3).

4. `\diagLegenditem{}{}` adds one item to the legend: #1 is text, #2 is color. A rectangle of the color #2 will be drawn and the text #1 printed next to it.

5. Parameters to be changed using `\renewcommand`:

   - `\legendShadow` (default 0) - can be 1(=show the legend shadow) or 0(=do not); is switched on or off automatically if the frame is 3D or 2D, but can be adjusted by the final user as well

- `\legendShadowColor` (default `legShCol`, where `\definecolor{legShCol}{gray}{.5}`)

6. Parameters to be changed using `\setlength`:
   - `\legenditemheight` (default 7mm) - see Fig.2.5
   - `\legendspacer` (default 3mm) - see Fig.2.5
   - `\legendRectWidth` (default 6mm) is width of the colored rectangle; see Fig.2.5
   - `\legendRectHeight` (default 4mm) is height of the colored rectangle; see Fig.2.5

7. Styles (to be changed by `\newpsstyle`)
   - `diagLegenditemstyle` is a style for the rectangle, which appears left to the legend text; default `fillstyle=solid,fillcolor=white,linecolor=white` is not supposed to be changed directly (use `diagLegendoptions`), although it's up to you.
   - +diagLegendframetyle+ is a style for the bounding box around the legend; default `fillstyle=none,linecolor=black,linewidth=0.5pt` is not supposed to be changed directly (use `diagLegendoptions`), although it's up to you.

3.5 3D effects

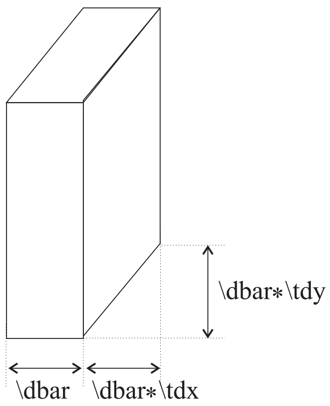


Figure 3.2: "Depth" of the 3D-effect

"Depth" of the 3D effect is controlled by `\tdx` and `\tdy` parameters (default - `\tdx`=0.4 and `\tdy` is 5% of the diagram height). They determine the "depth" in units of the bar width `\dbar`, Fig.3.2. Use `\renewcommand` to change them. Note, that the `\renewcommand` is to be issued *after* the `\begindiagram` statement.

Index