Creating Tufte-style bar charts and scatterplots using PGFPlots

Juernjakob Dugge

Abstract

In this article I describe how to use PGFPlots to create bar charts and scatterplots in the style described by Edward Tufte in *The Visual Display of Quantitative Information* [1].

I demonstrate how to implement *range frames*, which are axis lines drawn only over the range of the data points, and *dot-dash plots*, which are scatterplots with tick marks representing the marginal distribution of the data.

1 Bar chart

(I adapted this section from my `latex-community.org` post on this topic, namely `latex-community.org/know-how/437-tufte-charts`.)

Assume we have numerical data in a data file called `dataA.csv` that looks like this:

```
 1,  8.5
 2, 12
 3,  6.5
 4,  7
 5,  3
 6, 17.5
 7, 13
 8,  8.5
 9,  6
10, 11
11,  5
12, 10
```

Creating a bar chart of this data using PGFPlots is simple. All we have to do is put the following code at the point where we want the chart to appear:

```
\begin{tikzpicture}
  \begin{axis}[ybar]
    \addplot table [col sep=comma] {dataA.csv};
  \end{axis}
\end{tikzpicture}
```

The result is shown in Figure 1.

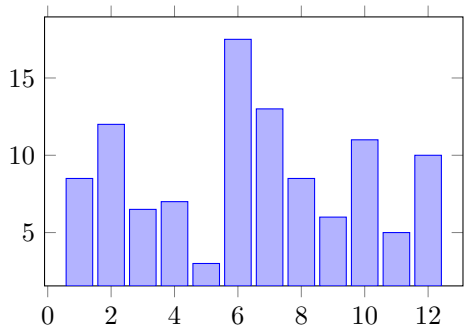

Figure 1: Bar chart using the default settings.

1.1 Bar colour

In his book, Tufte uses a particular shade of yellowish gray for the bars. Custom colours for use in PGFPlots can be defined using the `\definecolor` macro provided by the `xcolor` package, which is automatically loaded by PGFPlots. The colour used in Tufte's book can be made available using.

```
\definecolor{tufte1}{rgb}{0.7,0.7,0.55}
```

The most straightforward way to fill the bars of the plot using this colour and to disable the outlines of the bars is to add the keys `fill=tufte1, draw=none` to the optional argument of the `\addplot` command. Alternatively, we can set a `cycle list` consisting of these options:

```
cycle list={
  fill=tufte1, draw=none\\
}
```

That way, no options have to be supplied to the `\addplot` command.

1.2 Bar width

By default, the bars in PGFPlots are thicker than those in Tufte's plot. The width of the bars is controlled using the `bar width` key, which can be specified in absolute units (like `5mm`) or, since PGFPlots version 1.7, in terms of axis units. The latter requires setting the key `compat=1.7` or higher.

Using axis units to specify the bar width has the advantage that the widths of the bars will shrink as more data points are added, avoiding overlap between the bars. Sometimes, though, it might be desirable to keep the widths of the bars constant and instead increase the overall width of the plot as the number of data points increases. This can be achieved by using an absolute width for the bars, and specifying the length of the x unit vector in terms of the bar width:

```
bar width=2mm,
x=1.5*\pgfkeysvalueof{/pgfplots/bar width}
```

would make the bars 2 mm wide with a 1 mm gap between neighbouring bars, and the axis would grow or shrink horizontally to fit all the bars.

1.3 Grid lines

In Tufte's plot, there are gaps in the bars at regular intervals instead of conventional grid lines in the background of the bars. This can be simulated in PGFPlots by placing white horizontal grid lines on top of the bars by setting

```
ymajorgrids,
grid style=white,
axis on top
```

1.4 Axes

In Tufte's plot, the y axis is not drawn. In PGFPlots, an axis can be hidden using the key `hide y axis`. However, this key also deactivates the tick labels. In our case, since we only want to make the axis line invisible, we can instead set its opacity to zero using `y axis line style={opacity=0}`.

The x axis line in Tufte's plot is aligned with the first and last bar. This means that the x axis needs to run from the left edge of the first bar to the right edge of the last bar. In PGFPlots, this can be achieved by setting the padding of the x axis to half the bar width using

```
enlarge x limits={
  abs=0.5*\pgfkeysvalueof{/pgf/bar width}
}
```

where `abs` indicates that the padding is specified in terms of absolute units and not in axis units.

Since we only want an axis line at the bottom of the plot and not on the top, we set

```
axis x line*=bottom
```

By using the starred version of the key, no arrow tip is added to the axis line.

There are no tick labels on the x axis in Tufte's plot. In most real applications this would not be recommended, but in order to recreate Tufte's plot as faithfully as possible, let's go ahead and switch the labels off using `xtick=\empty`.

The y tick labels are expressed as percentages. We can specify how to print the tick labels in PGFPlots using the `yticklabel` key:

```
yticklabel=\pgfmathprintnumber{\tick}\,\%
```

The code passed to `yticklabel` is executed for every tick label. The `\tick` macro contains the current tick value, which is printed in a consistent format by `\pgfmathprintnumber`. After the tick value, we add a thin space (`\,`) and the percent sign, which has to be escaped using a backslash to distinguish it from its use as the comment character.

Note that there is also a `yticklabels` key (with a trailing `s`). This is used to provide a comma-separated *list* of tick labels, whereas the `yticklabel` key is used to provide a *pattern* for printing the labels, typically based on the tick value.

Finally, we can switch off the tick marks with

```
major tick length=0pt
```

1.5 Creating an axis style

There are different ways of activating all these options. The keys can be directly added to the optional argument of an `axis` environment, in which case they only apply to that axis.

They can also be activated globally using

```
\pgfplotsset{
  ⟨keys⟩
}
```

which applies the keys to all `axis` environments that follow.

Lastly, we can create a new PGF `style` that acts as a container for the keys, and apply that new `style` to the `axis` environment:

```
\pgfplotsset{
  tufte bar/.style={
    ⟨key1⟩,
    ⟨key2⟩,
    ...
  }
}
\begin{axis}tufte bar
```

Grouping options using `style`s is a very useful technique, as it helps to keep the code readable and maintainable.

Putting all the keys described above into a style and applying that style to the axis in the first listing results in the plot shown in Figure 2.

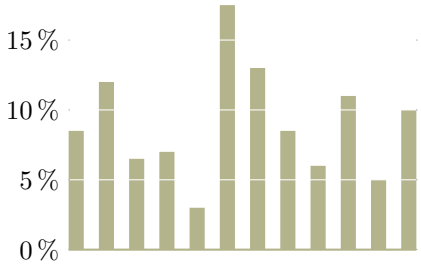

Figure 2: Bar chart in the style of Edward Tufte

2 Scatterplot

A scatterplot can be created in PGFPlots just as easily as a bar chart. A simple plot of some random data points like the one shown in Figure 3 can be created using

```
\begin{tikzpicture}
  \begin{axis}[
    only marks,
    domain=1:10
  ]
    \addplot
       ({cos(rnd r)*x+rnd},{(rnd+1)*x+rnd});
  \end{axis}
\end{tikzpicture}
```

The key `only marks` instructs PGFPlots not to draw connecting lines between the data points.

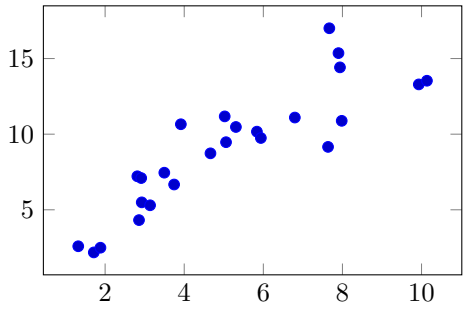

Figure 3: Default scatterplot

2.1 Basic scatterplot adjustments

With only a few options, we can already get quite close to the scatterplots in Tufte's book. The plot marks are black and slightly smaller than in the default plot, so we set

```
cycle list={black},
mark size=1.5pt
```

(the default mark size is 2 pt).

The remaining adjustments are related to the plot axes.

2.2 Range frames

Instead of conventional axis lines, Tufte uses so-called *range frames*: The axis lines are not drawn over the entire extent of the plot, but only between the levels of the lowest and highest data points.

One way of creating range frames in PGFPlots is by erasing the standard axis lines and drawing lines of the required lengths using `\draw` commands. For this, we can make use of the fact that PGFPlots stores the lowest and highest data coordinate values in internal macros. These macros, called

```
\pgfplots@data@xmin
\pgfplots@data@xmax
...
```

can be made accessible by putting

```
\makeatletter
\let\pgfplotsdataxmin=\pgfplots@data@xmin
\let\pgfplotsdataxmax=\pgfplots@data@xmax
\let\pgfplotsdataymin=\pgfplots@data@ymin
\let\pgfplotsdataymax=\pgfplots@data@ymax
\makeatother
```

before the specification of the `\draw` command. Then the values can be referred to as `\pgfplotsdataxmin`, `\pgfplotsdataxmax`, and so on.

To draw the lines, we can use

```
\draw ({rel axis cs:0,0}
    -|{axis cs:\pgfplotsdataxmin,0})
  -- ({rel axis cs:0,0}
      -|{axis cs:\pgfplotsdataxmax,0});
```

This might look a bit intimidating at first, so let's go through it step by step. The basic command is `\draw (A) -- (B);`, which simply draws a straight line between points `A` and `B`.

In this case, `A` is defined as

```
({rel axis cs:0,0}
  -|{axis cs:\pgfplotsdataxmin,0})
```

This is a coordinate specification of the type `(C-|D)`, which describes the point located at the intersection of a horizontal line through `C` and a vertical line through `D`.

Here, `C` is `rel axis cs:0,0`, which is the point in the lower left corner of the axis, and `D` is

```
axis cs:\pgfplotsdataxmin,0
```

which is the point with an x component equal to that of the leftmost data point and a y component of zero.

If the x axis was at $y = 0$, we could simply use

```
\draw (axis cs:\pgfplotsdataxmin,0)
  -- (axis cs:\pgfplotsdataxmax,0);
```

Since that is not necessarily the case, though, we'll have to make use of the more complicated expression.

To automatically execute the `\draw` command at the end of the plot, we pass it to the axis like this:

```
after end axis/.code={
  \draw ({rel axis cs:0,0}
      -|{axis cs:\pgfplotsdataxmin,0})
    -- ({rel axis cs:0,0}
      -|{axis cs:\pgfplotsdataxmax,0});
  \draw ({rel axis cs:0,0}
      |-{axis cs:0,\pgfplotsdataymin})
    -- ({rel axis cs:0,0}
      |-{axis cs:0,\pgfplotsdataymax});
}
```

To complete the plot style, only a couple of minor adjustments are needed.

Erase the default axis lines using

```
axis line style={opacity=0}
```

Only show the tick marks on the left and bottom edge of the plot by setting

```
tick pos=left
```

And finally, align the tick marks on the outside of the plot area using

```
tick align=outside
```

Wrapping all these keys in a new style and applying that style to the plot shown in Figure 3 results in the plot shown in Figure 4.

2.3 Dot-dash plot

Another technique used by Edward Tufte is the *dot-dash plot*, which is a combination of a conventional

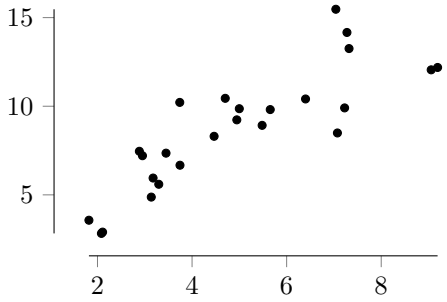

Figure 4: Scatterplot with range-frame

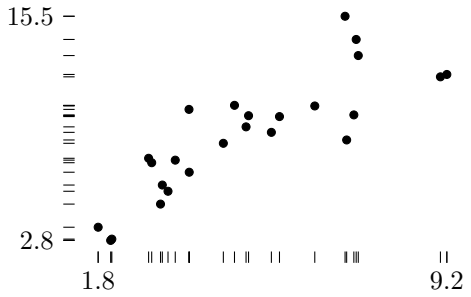

Figure 5: Dot-dash plot

scatterplot (the dots) and plots of the marginal distributions of the data points in the form of short lines along the outside of the plot (the dashes).

This can be implemented in PGFPlots surprisingly easily. By default, the tick marks are placed at regular intervals along the axis. By specifying

```
xtick=data,
ytick=data
```

tick marks are placed at the data points' locations.

Removing the axis lines and tick labels, and making the tick marks black and a bit longer is all it takes to create a basic dot-dash plot:

```
axis line style={opacity=0},
xticklabels={},
yticklabels={},
tick style=black
```

To aid the viewer in reading the plot, we can label the first and last of the tick marks. We can do this by again using our macros that store the limits of the data. PGFPlots makes it possible to highlight some tick positions by placing additional tick marks that can be specified using

```
extra x ticks={
  \pgfplotsdataxmin,
  \pgfplotsdataxmax
}
```

These extra ticks can be formatted independently from the standard ticks by specifying the required keys in

```
extra tick style={
  ⟨options⟩
}
```

In this case, we need to reactivate the tick labels for the extra ticks. By setting

```
extra tick style={
  xticklabel={\pgfmathprintnumber[
    fixed,
    fixed zerofill,
    precision=1
  ]{\tick}},
  yticklabel={\pgfmathprintnumber[
    fixed,
    fixed zerofill,
    precision=1
  ]{\tick}},
  scaled ticks=false
}
```

the values of the outer ticks will be printed as fixed point numbers rounded to one decimal digit. Setting `scaled ticks=false` is necessary when the number formatting style is explicitly set to `fixed`. Otherwise, PGFPlots would print a separate scaling factor when the axis contains very large numbers.

Applying all these options to the plot results in the plot shown in Figure 5.

3 Conclusion

In this article, I aimed to demonstrate the flexibility of PGFPlots by recreating plots from Edward Tufte's *The Visual Display of Quantitative Information*.

While some of the techniques used in this article used internal PGFPlots macros, no alteration of the code was necessary to implement reasonably advanced features like range frames or dot-dash plots.

I hope that this article will succeed in encouraging some of the readers to try their hand at more intricate plot customisations of their own. Once the necessary options and values have been found, the `style` feature of the `pgfkeys` key-value framework used by PGFPlots makes it very easy to reuse plot styles. This helps in creating plots with a consistent appearance with very little effort.

Acknowledgments

I would like to thank Dr. Christian Feuersänger for his very helpful review of this manuscript, and for his continued excellent work on PGFPlots.

References

[1] Edward R. Tufte. *The visual display of quantitative information*. Graphics Press, Cheshire, Conn, 2nd edition, 2001.

Appendix: Complete styles

Make the extreme values available and define the colour:

```
\makeatletter
\let\pgfplotsdataxmin=\pgfplots@data@xmin
\let\pgfplotsdataxmax=\pgfplots@data@xmax
\let\pgfplotsdataymin=\pgfplots@data@ymin
\let\pgfplotsdataymax=\pgfplots@data@ymax
\makeatother

\definecolor{tufte1}{rgb}{0.7,0.7,0.55}
```

3.1 Bar chart

```
\pgfplotsset{
  tufte bar/.style={
    ybar,
    axis line style={draw opacity=0},
    xtick=\empty,
    ymin=0,
    bar width=2mm,
    x=2*\pgfkeysvalueof{/pgf/bar width},
    ymajorgrids,
    grid style=white,
    axis on top,
    major tick length=0pt,
    cycle list={
      fill=tufte1, draw=none\\
    },
    enlarge x limits={
      abs=0.5*\pgfkeysvalueof{/pgf/bar width}
    },
    axis x line*=bottom,
    x axis line style={
      draw opacity=1,
      tufte1,
      thick
    },
    yticklabel=\pgfmathprintnumber{\tick}\,\%
  }
}
```

3.2 Basic scatterplot

```
\pgfplotsset{
  tufte scatter/.style={
    only marks,
    cycle list={black, gray!50},
    axis lines*=left,
    mark size=1.5
  }
}
```

3.3 Range frame

```
\pgfplotsset{
  range frame/.style={
    tick align=outside,
    axis line style={opacity=0},
    after end axis/.code={
      \draw ({rel axis cs:0,0}
          -|{axis cs:\pgfplotsdataxmin,0})
        -- ({rel axis cs:0,0}
          -|{axis cs:\pgfplotsdataxmax,0});
      \draw ({rel axis cs:0,0}
          |-{axis cs:0,\pgfplotsdataymin})
        -- ({rel axis cs:0,0}
          |-{axis cs:0,\pgfplotsdataymax});
    }
  }
}
```

3.4 Dot-dash plot

```
\pgfplotsset{
  dot dash plot/.style={
    tufte scatter
    axis line style={opacity=0},
    tick style={thin, black},
    major tick length=0.15cm,
    xtick=data,
    xticklabels={},
    ytick=data,
    yticklabels={},
    extra x ticks={
      \pgfplotsdataxmin,
      \pgfplotsdataxmax
    },
    extra y ticks={
      \pgfplotsdataymin,
      \pgfplotsdataymax
    },
    extra tick style={
      xticklabel={\pgfmathprintnumber[
        fixed,
        fixed zerofill,
        precision=1
      ]{\tick}},
      yticklabel={\pgfmathprintnumber[
        fixed,
        fixed zerofill,
        precision=1
      ]{\tick}}
    }
  }
}
```

◇ Juernjakob Dugge
Schwaerzlocher Str. 64
72070 Tuebingen
Germany
juernjakob (at) dugge dot de