TikZ annotations for ggplots with the ggtikz package

September 12, 2025

Contents

1 Prerequisites

1.1 LaTeX side

As the name implies, `ggtikz` requires `tikz`, which must be loaded in the document's preamble. Furthermore, the `calc` tikz library is required.

Thus, the preamble must contain:

```
\usepackage{tikz}
\usetikzlibrary{calc}
```

1.2 R side

The `tikzDevice` package is required to render plots and `ggtikz` annotations to the tikz format. We also have to make some base plots, using `ggplot2`.

Here, we set the graphics device to `tikz` – `ggtikz` does not work with any other graphics device!

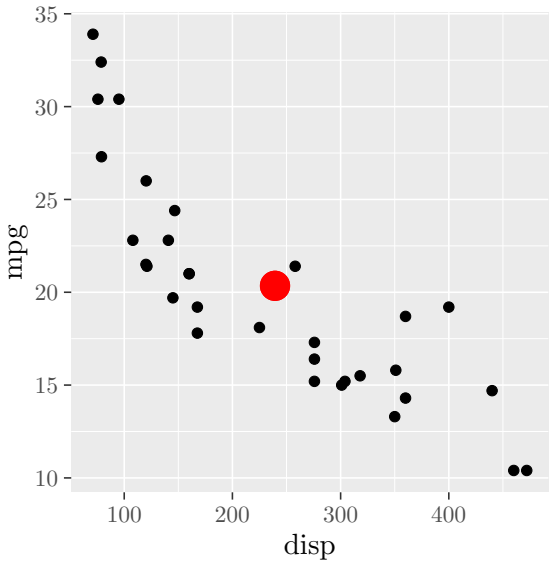
```
library(knitr)
library(ggplot2)
library(ggtikz)
opts_chunk$set(
    dev = "tikz",
    error = TRUE,
    cache = FALSE,
    external = TRUE,
    fig.path = "example-vignette-figures/",
    fig.width = 3,
    fig.height = 3,
    fig.align = "center"
)
```

2 Basic usage with `ggtikz()`

For simple one-step annotations, the `ggtikz` helper function is available.

It accepts a ggplot object as its first argument. Further arguments are passed on to `ggtikzAnnotation` (see section 3).

```
p <- ggplot(mtcars, aes(disp, mpg)) + geom_point()
ggtikz(p, "\\fill[red] (0.5,0.5) circle (2mm);", xy="plot")
```



3 Advanced usage with canvases and annotations

With `ggtikz()`, only a single annotation can be added to a plot. If multiple annotations are needed, then we first need to create a `ggtikzCanvas()`, to which one or more `ggtikzAnnotation()` can be added.

3.1 Single-panel plots

Let's create a single-panel plot for annotation.

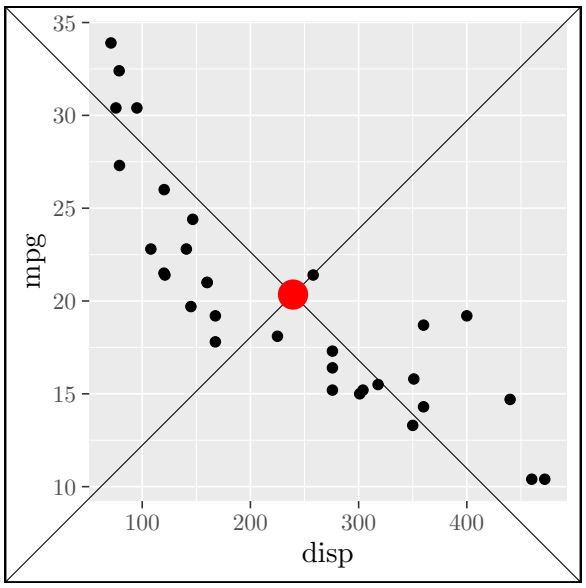
```
p <- ggplot(mtcars, aes(disp, mpg)) +
    geom_point() +
    theme(plot.background=element_rect(color = "black", linewidth = 1))
```

We can then set up an annotation canvas and add tikz annotations. Note that first, we print the base plot to the device [1], and then the annotation canvas. The annotation canvas does not take care of drawing the annotated plot (the `ggtikz()` helper does handle this with the `draw = TRUE` parameter).

3.1.1 Annotation relative to the whole plot

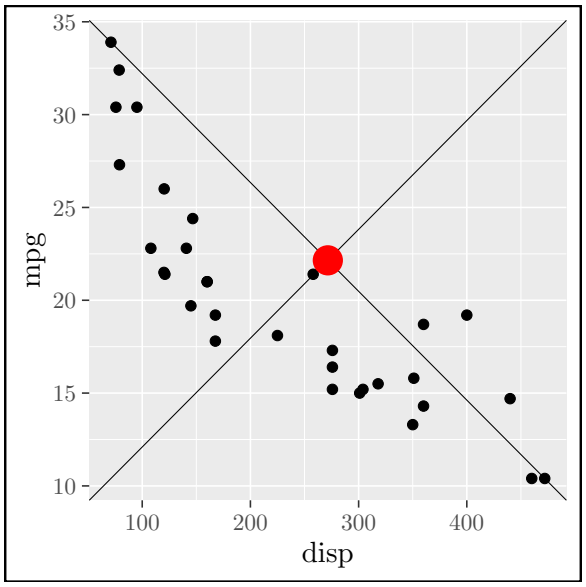
[1] no explicit calls to `tikz()` and `dev.off()` are needed, because knitr opens and closes the device automatically.

```
canvas <- ggtikzCanvas(p)
annotation <- ggtikzAnnotation(
    "
    \\draw (0,0) -- (1,1);
    \\draw (0,1) -- (1,0);
    \\fill[red] (0.5,0.5) circle (2mm);
    ",
    xy = "plot"
)
p                       # first draw the plot
canvas + annotation     # then draw the annotations
```



3.1.2 Annotation relative to the panel

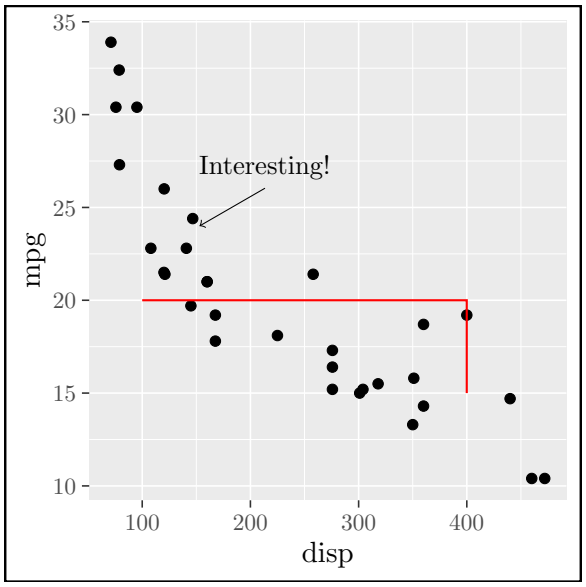
```
canvas <- ggtikzCanvas(p)
annotation <- ggtikzAnnotation(
    "
    \\draw (0,0) -- (1,1);
    \\draw (0,1) -- (1,0);
    \\fill[red] (0.5,0.5) circle (2mm);
    ",
    xy = "panel",
    panelx = 1, panely = 1
)
p
canvas + annotation
```



3.1.3 Annotation relative to data coordinates

In addition to unitless tikz coordinates, you can also use absolute lengths, such as the 1 cm in the example below.

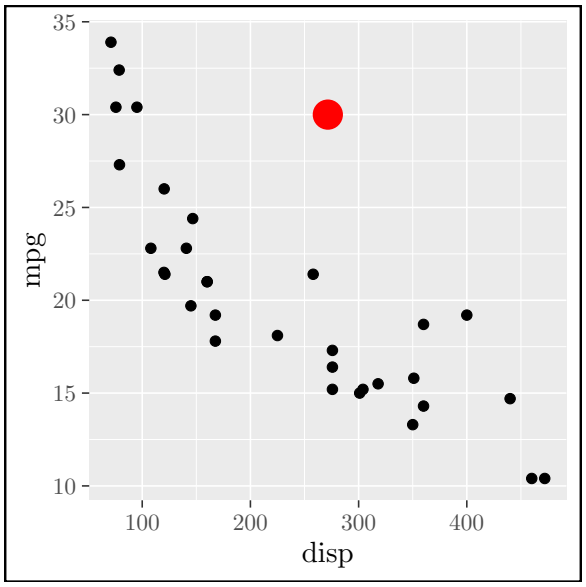
```
canvas <- ggtikzCanvas(p)
annotation <- ggtikzAnnotation(
    "
    \\draw[thick,red] (100,20) -| (400,15);
    \\draw[<-] (153,24) -- ++(30:1cm) node[at end, anchor=south]
        {Interesting!};
    ",
    xy = "data",
    panelx = 1, panely = 1
)
p
canvas + annotation
```



3.1.4 Mixing panel and data references

The reference frames for x and y coordinates can be separately assigned as `data` or `panel`. However, note that the `plot` reference frame must be given for both x and y directions (with the `xy` argument), and cannot be mixed!

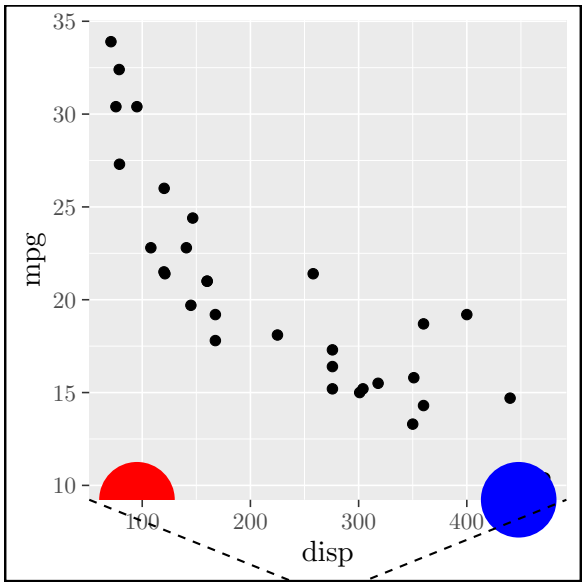
```
canvas <- ggtikzCanvas(p)
annotation <- ggtikzAnnotation(
   "\\fill[red] (0.5,30) circle (2mm);",
   x = "panel", y = "data",
   panelx = 1, panely = 1
)
p
canvas + annotation
```



3.1.5 Turning off clipping

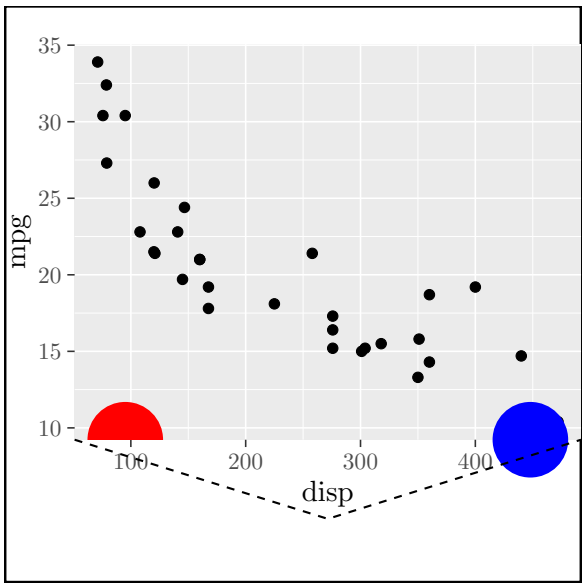
It is possible to turn off clipping for annotations, in order to draw outside of the plot area.

```
canvas <- ggtikzCanvas(p)
annotation_clip <- ggtikzAnnotation(
   "\\fill[red] (0.1,0) circle (5mm);",
   xy = "panel",
   panelx = 1, panely = 1
)

annotation_unclip <- ggtikzAnnotation(
   "\\fill[blue] (0.9,0) circle (5mm);",
   xy = "panel",
   panelx = 1, panely = 1,
   clip = "off"
)

annotation_unclip2 <- ggtikzAnnotation(
   "\\draw[thick, dashed] (0,0) -- (0.5,-0.2) -- (1,0);",
   xy = "panel",
   panelx = 1, panely = 1,
   clip = "off"
)
p
canvas + annotation_clip + annotation_unclip + annotation_unclip2
```



However, note that the surrounding plot area is not automatically unclipped to accommodate for the annotations. This can be alleviated manually by increasing the plot margins.

```
p + theme(plot.margin = margin(t=0.5, b = 1, unit = "cm"))
canvas + annotation_clip + annotation_unclip + annotation_unclip2
```

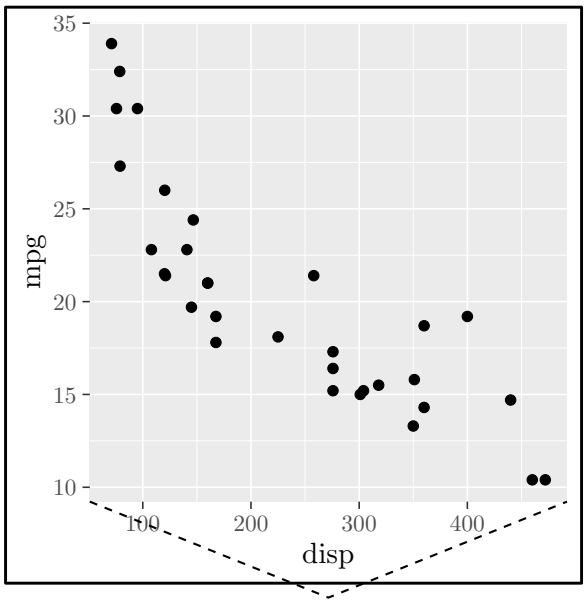


Alternatively, `ggtikz` comes with a knitr hook to automatically unclip TikZ files:

```
set_ggtikz_unclip_hook()
```

Now clipping can be disabled for chunks with the *chunk option* `unclip = TRUE` – however, this only works in conjunction with `external = FALSE`. If the option `external = TRUE`, then the resulting file is immediately compiled to pdf and not accessible for further post-processing.

```
# chunk options: external=FALSE, unclip=TRUE
p
canvas + annotation_unclip2
```



Unset the hook to restore the default clipping behavior:

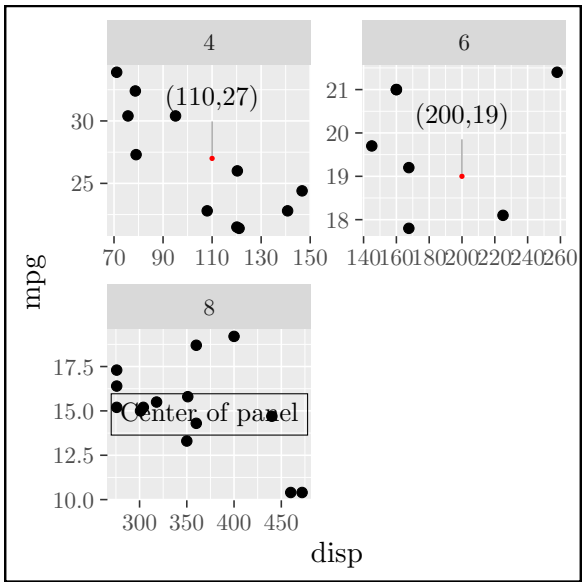
```
unset_ggtikz_unclip_hook()
```

3.2 Multi-panel plots: wrap

```
p_wrap <- p + facet_wrap(~cyl, scales="free", ncol=2)
```

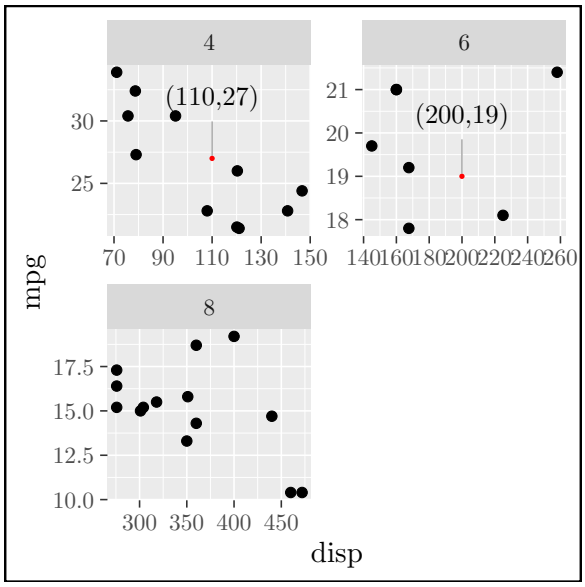
3.2.1 Annotations in separate panels, relative to data or panel coordinates

```
canvas <- ggtikzCanvas(p_wrap)

# Relative to data coordinates
annotation1 <- ggtikzAnnotation(
    "
    \\node[pin={90:(110,27)}, circle, fill=red,
        inner sep=0, outer sep=0, minimum size=2pt]
        at (110,27)
        {};
    ",
    xy = "data",
    panelx = 1, panely = 1
)

# Relative to data coordinates
annotation2 <- ggtikzAnnotation(
    "
    \\node[pin={90:(200,19)}, circle, fill=red,
    inner sep=0, outer sep=0, minimum size=2pt]
    at (200,19)
    {};
    ",
    xy = "data",
    panelx = 2, panely = 1
)

# Relative to panel coordinates
annotation3 <- ggtikzAnnotation(
    "
    \\node[draw, anchor=center] at (0.5, 0.5)
        {Center of panel};
    ",
    xy = "panel",
    panelx = 1, panely=2
)

p_wrap
canvas + annotation1 + annotation2 + annotation3
```



3.2.2 Annotations in separate panels, relative to data coordinates

```
canvas <- ggtikzCanvas(p_wrap)
annotation1 <- ggtikzAnnotation(
    "
    \\node[pin={90:(110,27)}, circle, fill=red,
        inner sep=0, outer sep=0, minimum size=2pt]
        at (110,27)
        {};
    ",
    xy = "data",
    panelx = 1, panely = 1
)
annotation2 <- ggtikzAnnotation(
    "
    \\node[pin={90:(200,19)}, circle, fill=red,
    inner sep=0, outer sep=0, minimum size=2pt]
    at (200,19)
    {};
    ",
    xy = "data",
    panelx = 2, panely = 1
)
p_wrap
canvas + annotation1 + annotation2
```



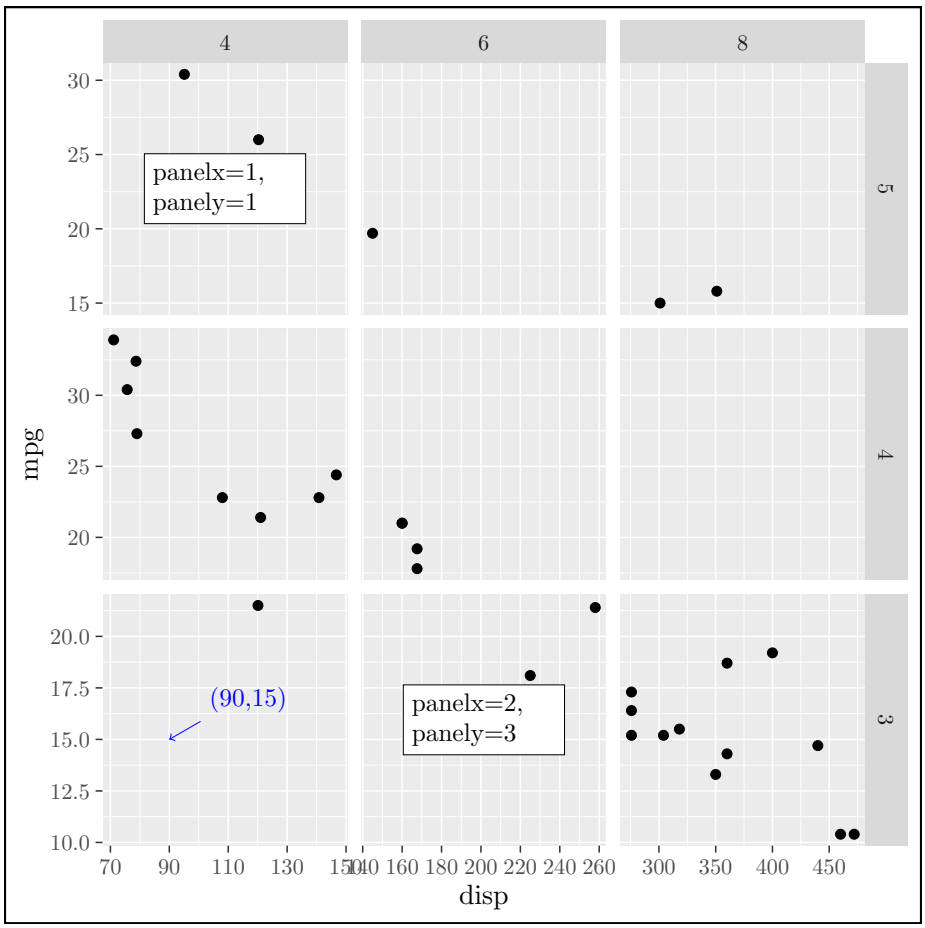
3.3 Multi-panel plots: grid

Annotations can also be made on individual panels of plots faceted with `facet_grid`.

```
p_grid <- p + facet_grid(gear~cyl, scales="free", as.table=FALSE)
```

```
canvas <- ggtikzCanvas(p_grid)
annot_grid1 <- ggtikzAnnotation(
    "\\node[fill=white, draw, text width=2cm] at (0.5,0.5)
        {panelx=1, panely=1};",
    xy = "panel",
    panelx = 1, panely = 1
)
annot_grid2 <- ggtikzAnnotation(
    "\\node[fill=white, draw, text width=2cm] at (0.5,0.5)
        {panelx=2, panely=3};",
    xy = "panel",
    panelx = 2, panely = 3
)
annot_grid3 <- ggtikzAnnotation(
    "
    \\draw[<-, blue] (90,15) -- ++(30:5mm)
        node [at end, anchor=south west] {(90,15)};
    ",
    xy = "data",
```

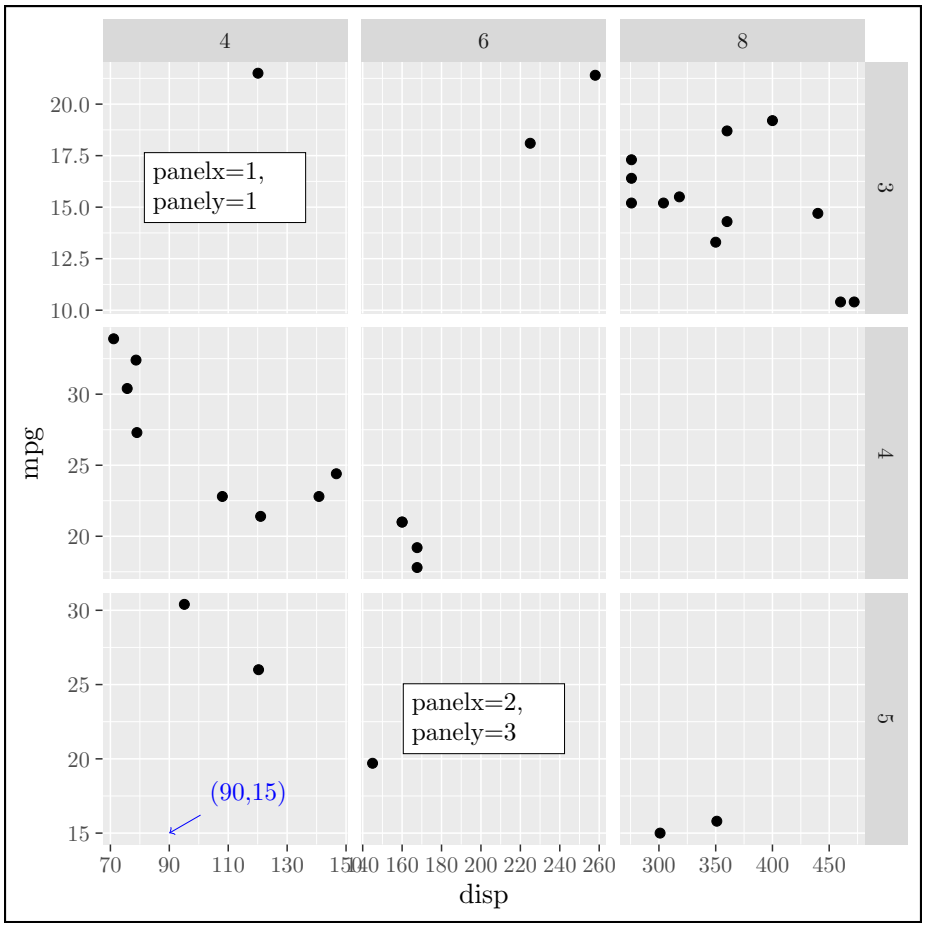
```
    panelx = 1, panely = 3
)
p_grid
canvas + annot_grid1 + annot_grid2 + annot_grid3
```



3.4 Re-using annotations

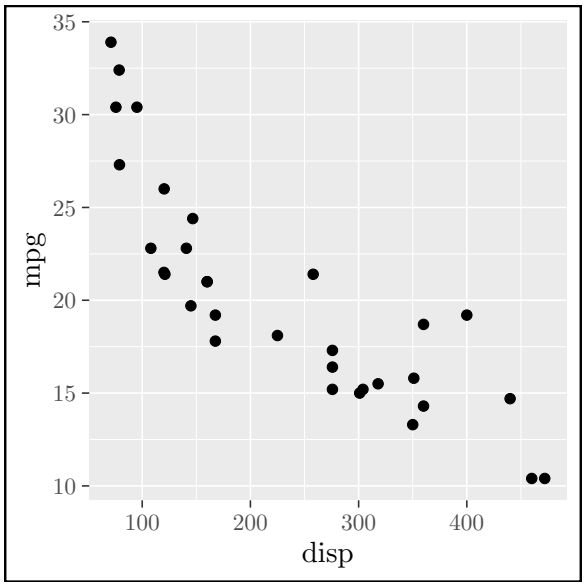
Annotations can be re-used between plots and `ggtikz` canvases. However, be aware that panel position specifications rely on the *visual position* of the panels, and *not on the value of the facet variables.*

```
p_grid2 <- p + facet_grid(gear~cyl, scales="free", as.table=TRUE)
canvas2 <- ggtikzCanvas(p_grid2)

p_grid2
canvas2 + annot_grid1 + annot_grid2 + annot_grid3
```



It is also not possible to add annotations to a plot for which the requested panels are not available.

```
p
```




```
canvas <- ggtikzCanvas(p)
canvas + annot_grid2

## Error in get_annotation_valid.ggtikzCanvas(self, ggtikzAnnotation):
Annotation wants to be placed in panelx = 2, but the plot only has
1.
```

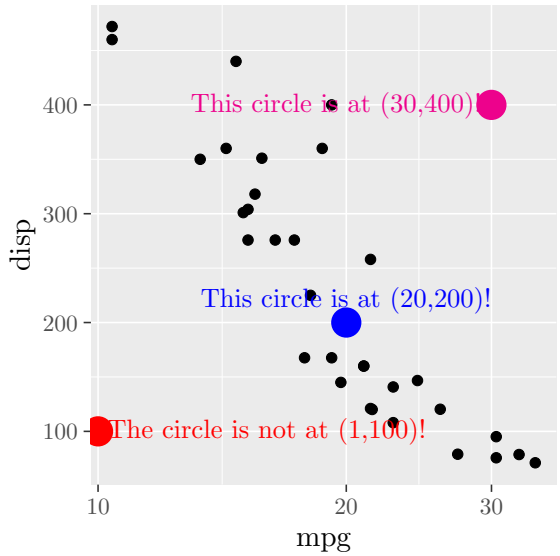
4 Transformed scales

TikZ coordinates can automatically be transformed to accommodate transformed scales, such as log-transformed ones. This is activated by setting `transform = TRUE` in the call to `ggtikzAnnotation`. With `transform = FALSE`, annotations made with data coordinates are out of place in log plots, but the transformation calculation can be done manually.

```
p_log <- ggplot(mtcars, aes(mpg, disp)) +
    geom_point() +
    scale_x_continuous(trans="log10")
```

```
canvas_log <- ggtikzCanvas(p_log)
# Untransformed coordinates: wrong position
annot_log <- ggtikzAnnotation(
    "\\fill[red] (1,100) circle (2mm);
    \\node[anchor=west, text=red] at (1, 100)
```

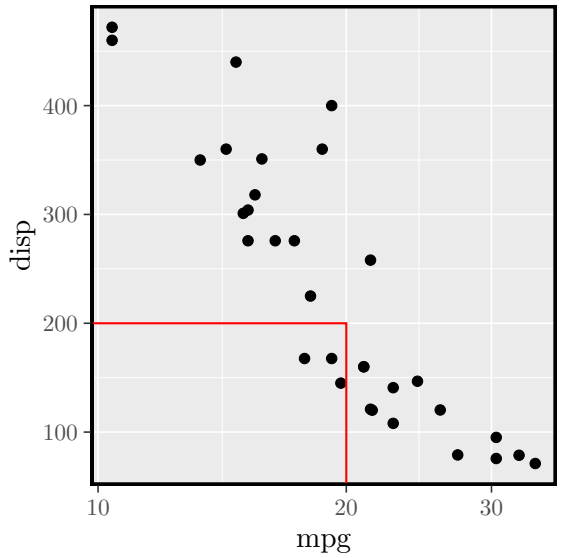
```
        {The circle is not at (1,100)!};
    ", xy = "data", transform = FALSE, panelx = 1, panely = 1
)

# Transformed coordinates: correct position
# The literal coordinate in the node text was wrapped in an \mbox
# LaTeX command to prevent automatic transformation -- it can't
# distinguish between coordinates which are supposed to be text,
# and actual coorinates.
annot_log2 <- ggtikzAnnotation(
    "\\fill[blue] (20,200) circle (2mm);
    \\node[anchor=south, text=blue] at (20, 200)
        {This circle is at (\\mbox{20,200})!};
    ", xy = "data", transform = TRUE, panelx = 1, panely = 1
)

# Untransformed coordinates, calculated manually by hand:
# correct position
annot_log3 <- ggtikzAnnotation(
    "\\fill[magenta] (1.477,400) circle (2mm);
    \\node[anchor=east, text=magenta] at (1.477, 400)
        {This circle is at (30,400)!};
    ", xy = "data", transform = FALSE, panelx = 1, panely = 1
)
p_log
canvas_log + annot_log + annot_log2 + annot_log3
```



5 Using *Inf* and *-Inf* in annotations

In `ggplot2`, `-Inf` and `Inf` can be used to refer to the edge of a panel. This is also possible in ggtikzAnnotations, by setting `replace_inf = TRUE`. `-Inf` and `Inf` will then be automatically replaced to refer to the left/bottom or right/top edge of a panel, respectively.

```
p_log_border <- p_log +
    theme(panel.border = element_rect(fill = NA, linewidth = 2))
canvas_log_border <- ggtikzCanvas(p_log_border)
annot_inf <- ggtikzAnnotation(
    "\\draw[red, thick] (-Inf,200) -| (20,-Inf);
    \\draw[green, thick] (-Inf,200) |- (20,-Inf);
    ", xy = "data", replace_inf = TRUE, panelx = 1, panely = 1
)
p_log_border
canvas_log_border + annot_inf
```

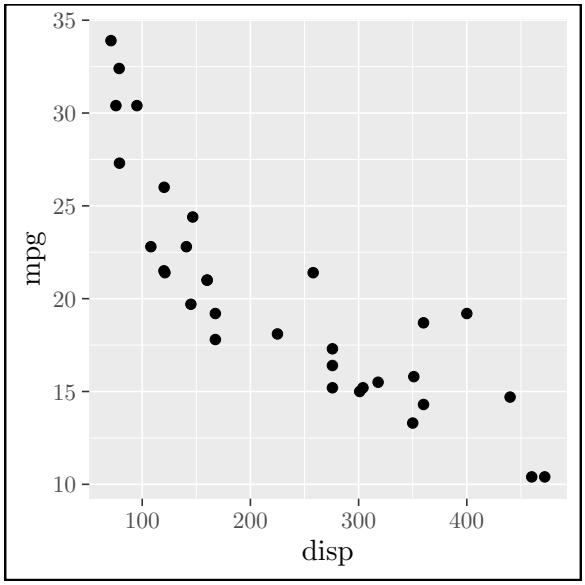


6 Using styles defined in the surrounding document

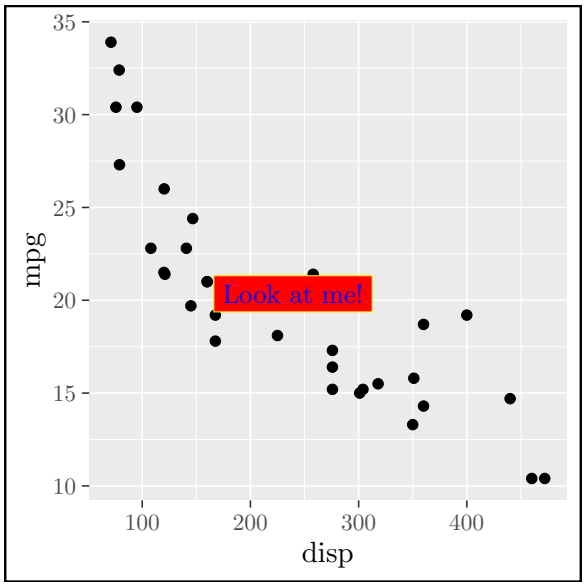
Annotations can access styles which are defined in the containing document before the relevant .tikz file is included, allowing you to re-use global styles. Note that by default, knitr sets the option `external` to `TRUE`. Therefore, tikz graphics are pre-compiled to pdf. In that case, the tikzDevice needs to know about these styles, or an error will occur during externalization.

```
\tikzset{loud/.style={
     draw=yellow,
     fill=red,
     text=blue
}}
```

```
p
```




```
canvas <- ggtikzCanvas(p)
styled_annot <- ggtikzAnnotation(
    "\\node[loud] at (0.5,0.5) {Look at me!};",
    xy = "plot"
)
p
canvas + styled_annot
```


35
30
25
20
15
10
mpg
Look at me!
100
200
300
400
disp