

```
library(knitr)
knitr::opts_chunk$set(fig.show = "hide",
  fig.path = 'atlas/',
  fig.width = 11,
  fig.height = 7,
  tidy = FALSE,
  #message = FALSE,
  out.width = "11in",
  out.height = "7in")
```

```
library(ggplot2)
library(scales)
library(grattan)

##
## Attaching package: 'grattan'
##
## The following object is masked from 'package:datasets':
##
##   Orange

library(tidyr)
library(dplyr)

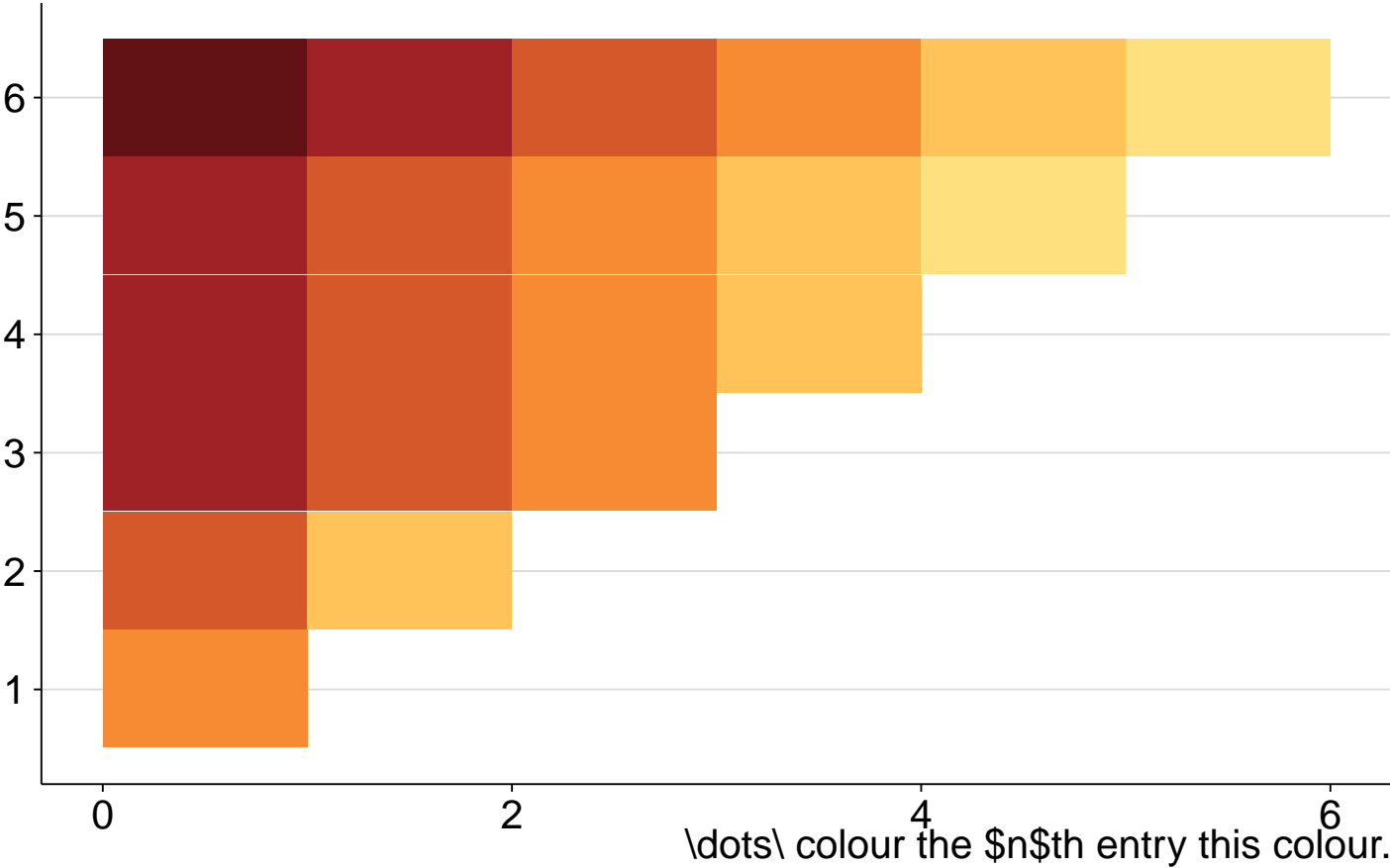
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:stats':
##
##   filter, lag
##
## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

library(magrittr)

##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:tidyr':
##
##   extract
```

```
.simpleCap <- function(x) {
  s <- strsplit(x, " ")[[1]]
  paste(toupper(substring(s, 1, 1)), substring(s, 2),
    sep = "", collapse = " ")
}
```

```
p <- grplot(NULL)
for (y in 1:6){
  for (x in 1:6){
    p <- p + annotate("rect",
                      xmin = x - 1, xmax = x,
                      ymin = y - 0.99, ymax = y,
                      fill = gpal(y)[x])
  }
}
p +
  scale_y_continuous("With this number of categories",
                     breaks = 0:6 + 0.5, labels = 1:7, limits = c(0,6)) +
  scale_x_continuous("\\dots\\ colour the $n$th entry this colour.")
```

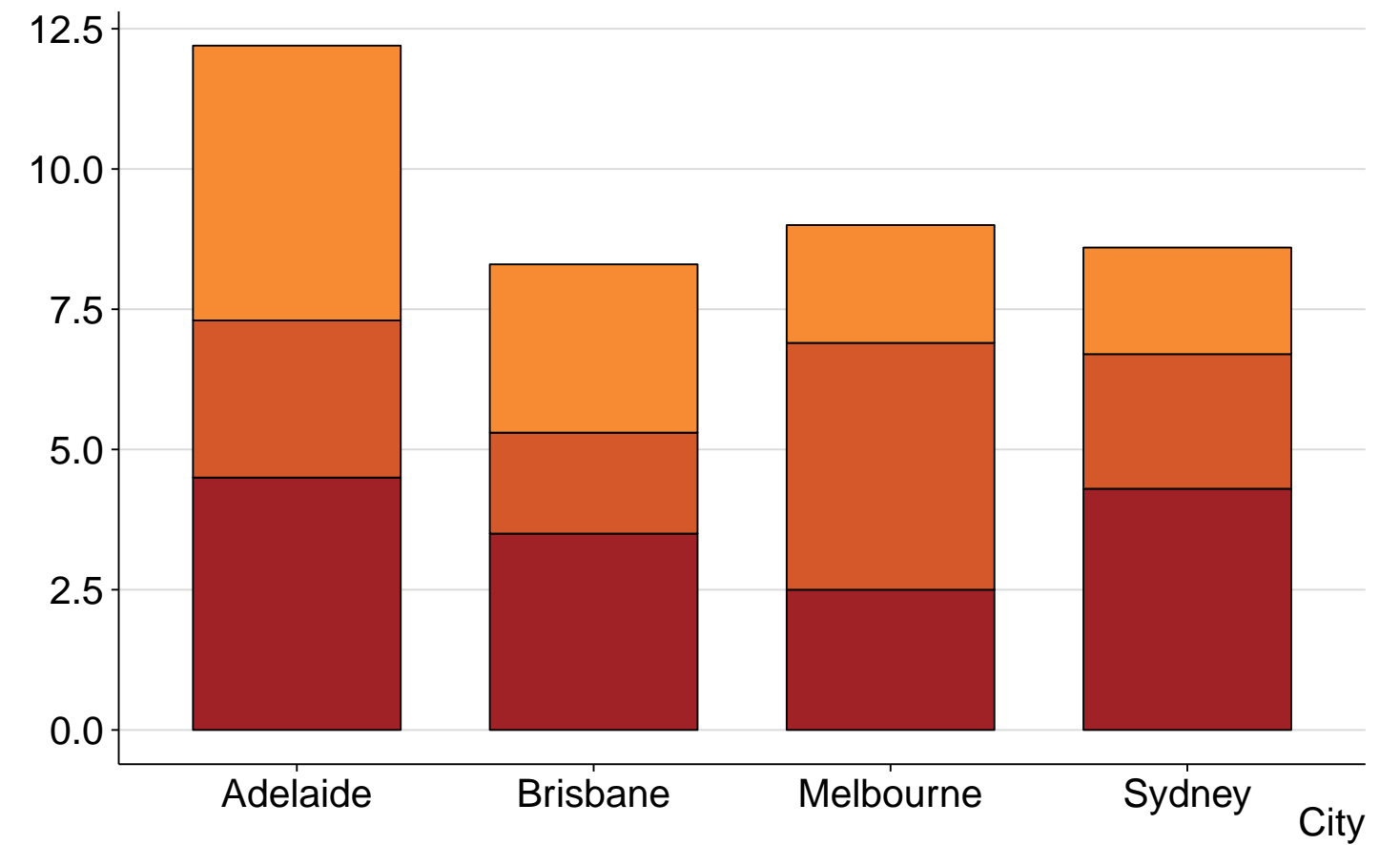


```
p <- grplot(NULL)
x <- 0
for (gray in c(217,174,130,87,43,0)/256){
  x <- x + 1
  p <- p + annotate("rect", xmin = x - 1, xmax = x, ymin = 0.5, ymax = 1.5,
                    fill = rgb(gray, gray, gray)) +
    coord_equal() +
    theme_void()
}
p
```



## 0.1 Bar charts

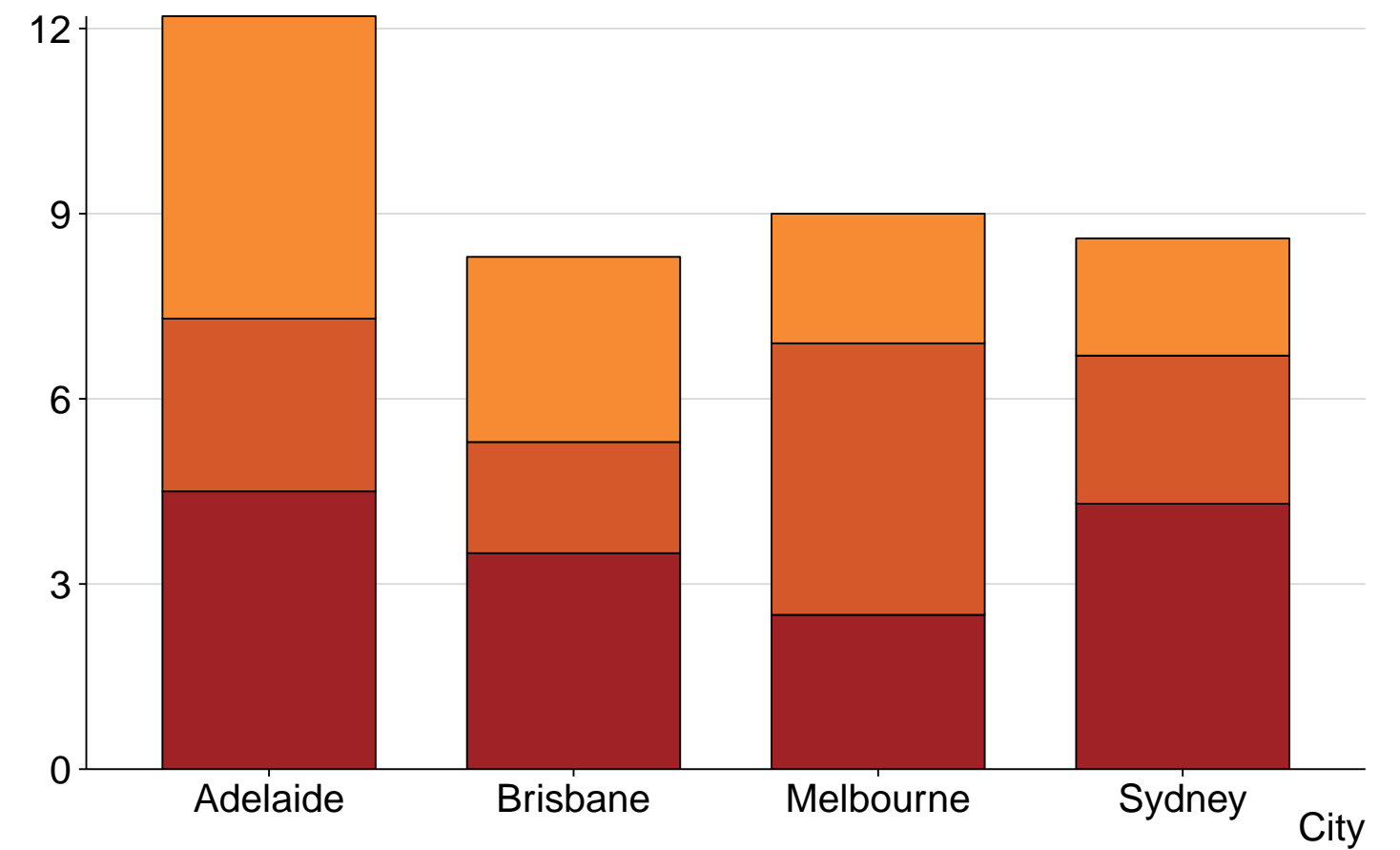
```
read.table(text ="
City  1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7)
```



### 0.1.1 Axes flush with data

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7) +

  # To adjust the spacing between the axes, use expand = c(0,0)
  scale_y_continuous(expand = c(0,0))
```

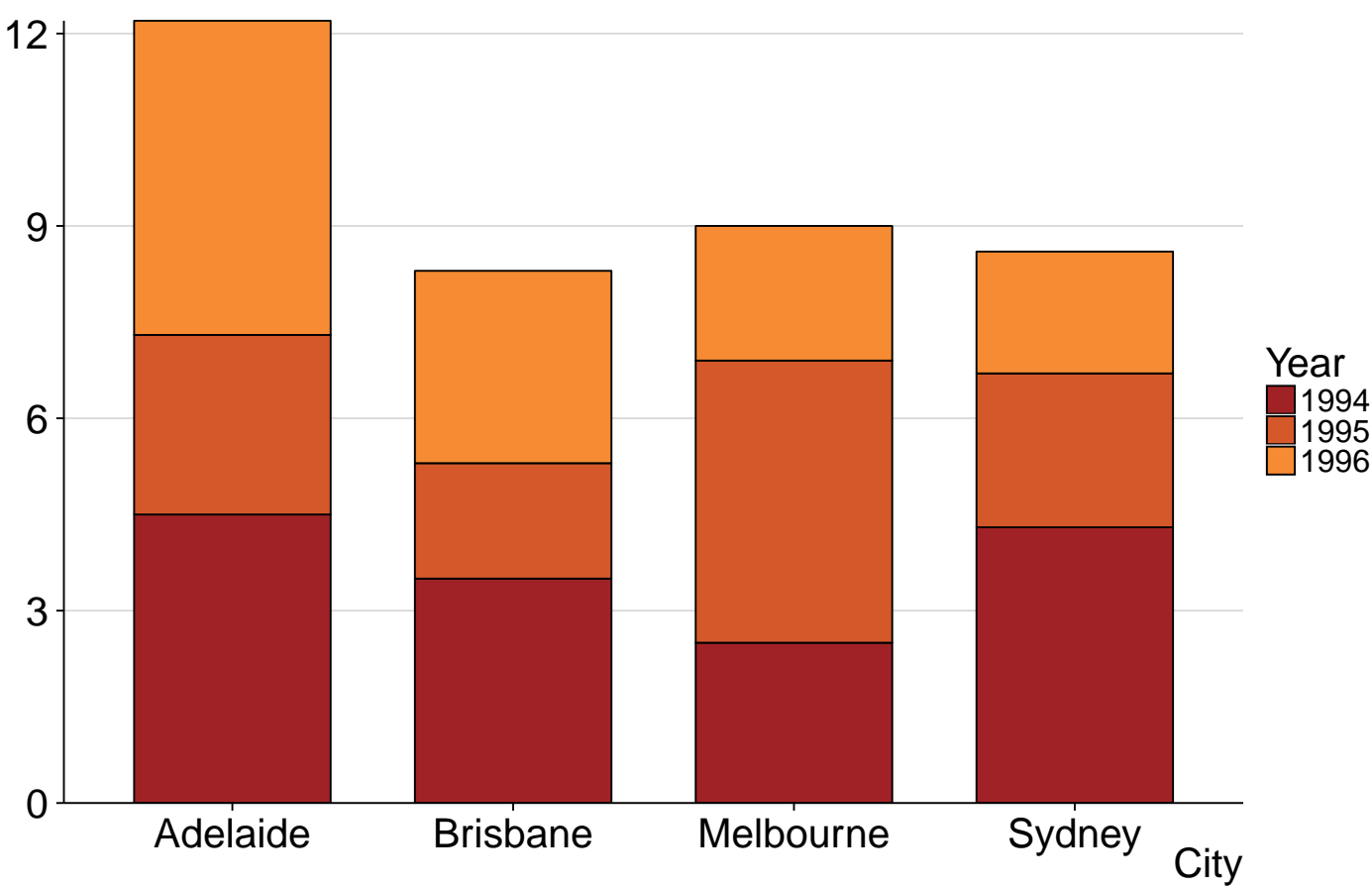


0.1.2 Add legend

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7) +

  # To adjust the spacing between the axes, use expand = c(0,0)
  scale_y_continuous(expand = c(0,0)) +

  # Include a legend:
  # Without the following, we get `factor(Year)` as the
  # legend title. This sets the legend title.
  guides(fill = guide_legend("Year")) +
  theme(legend.position = "right")
```

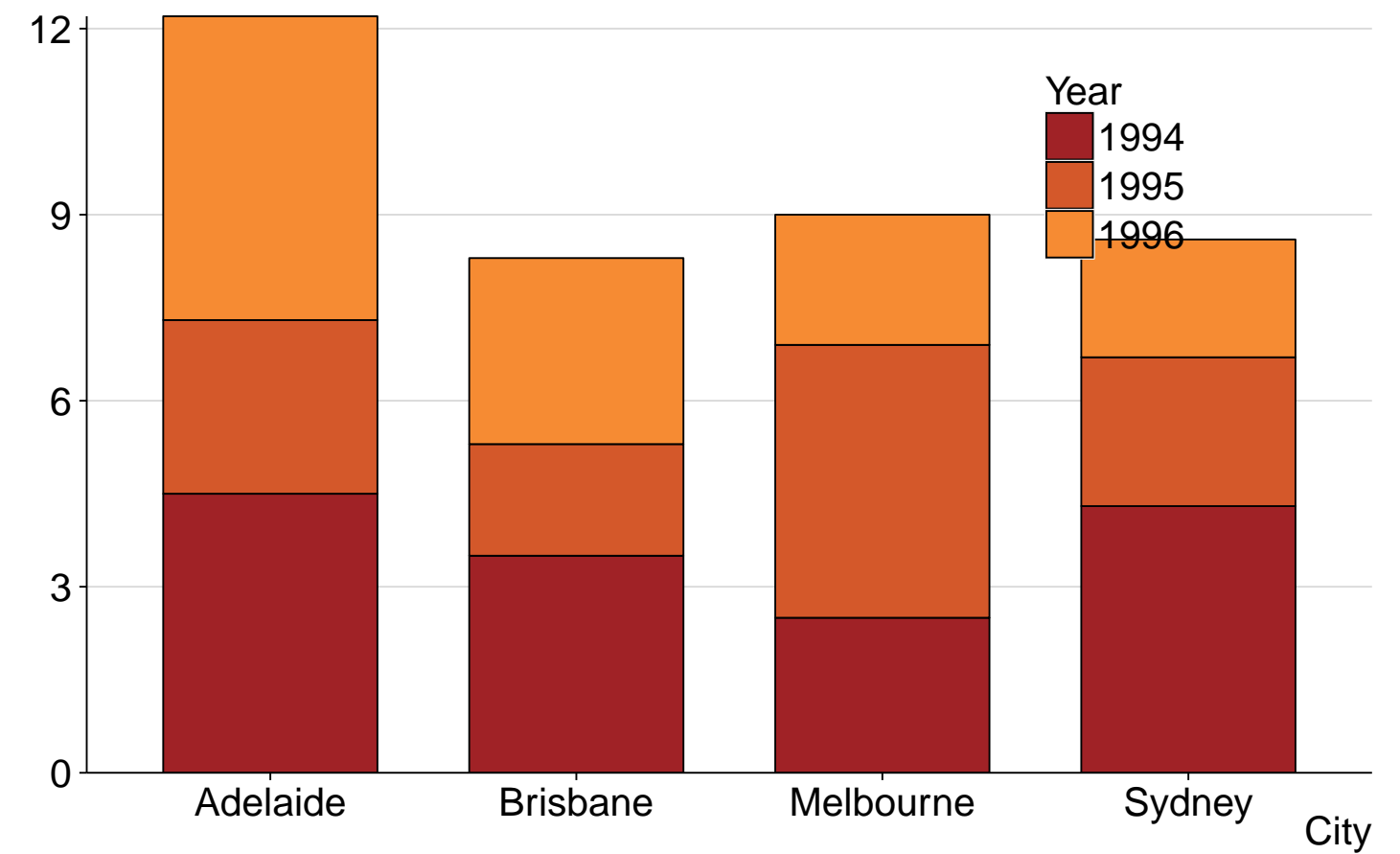


### 0.1.3 Add legend over plot

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7) +

  # To adjust the spacing between the axes, use expand = c(0,0)
  scale_y_continuous(expand = c(0,0)) +

  # Include a legend:
  # Without the following, we get `factor(Year)` as the
  # legend title. This sets the legend title.
  guides(fill = guide_legend("Year")) +
  # you can also adjust the legend position ranged
  # c(0,0) = southwest corner
  # c(1,0) = southeast corner
  # c(0,1) = northwest corner
  # c(1,1) = northeast corner
  # ranged between
  theme(legend.position = c(0.8, 0.8),
        # play with unit(<width> , "lines")
        legend.text = element_text(size = 23),
        legend.key.size = unit(2, "lines"))
```

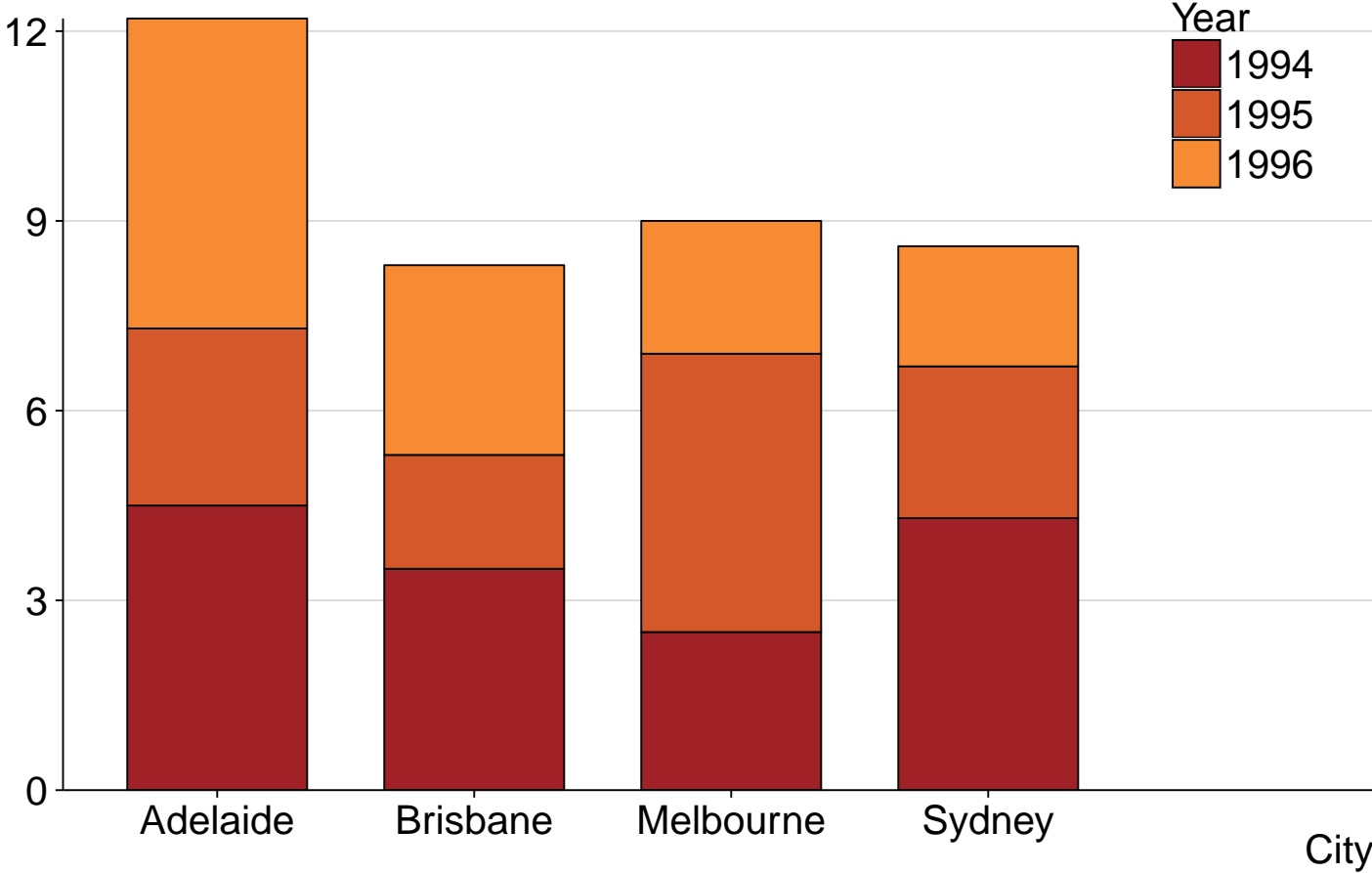


0.1.4 Adding space for legend

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7) +

  # To adjust the spacing between the axes, use expand = c(0,0)
  scale_y_continuous(expand = c(0,0)) +

  # Include a legend:
  # Without the following, we get `factor(Year)` as the
  # legend title. This sets the legend title.
  guides(fill = guide_legend("Year")) +
  # you can also adjust the legend position ranged
  # c(0,0) = southwest corner
  # c(1,0) = southeast corner
  # c(0,1) = northwest corner
  # c(1,1) = northeast corner
  # ranged between
  theme(legend.position = c(0.9, 0.9),
        # play with unit(<width> , "lines")
        legend.text = element_text(size = 23),
        legend.key.size = unit(2, "lines")) +
  #
  # you can use blank annotations to expand the axis
  annotate("blank",
    x = 5.5,
    y = NA_real_)
```





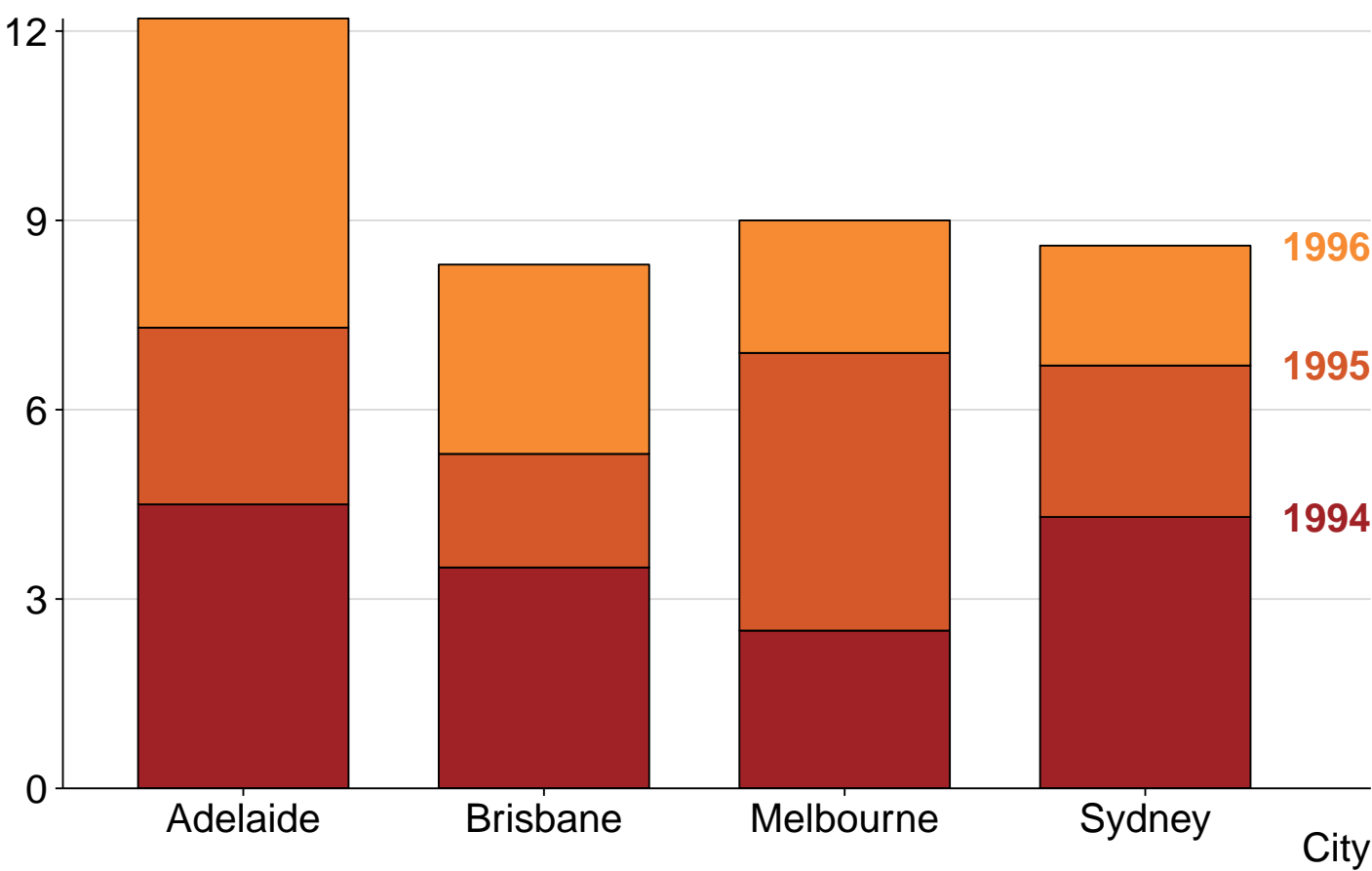
0.1.5 Coloured text as legend

```
read.table(text = "
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  #
  # add a label variable. Exploit the fact that if any
  # variable is NA then the text will not be plotted.
  mutate(text.label = ifelse(as.numeric(City) == max(as.numeric(City)),
                             as.character(Year),
                             NA_character_),
         text.x = as.numeric(factor(City)) + 0.75) %>%
  group_by(City) %>%
  mutate(text.y = cumsum(value)) %>%
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.7) +

  # To adjust the spacing between the axes, use expand = c(0,0)
  scale_y_continuous(expand = c(0,0)) +

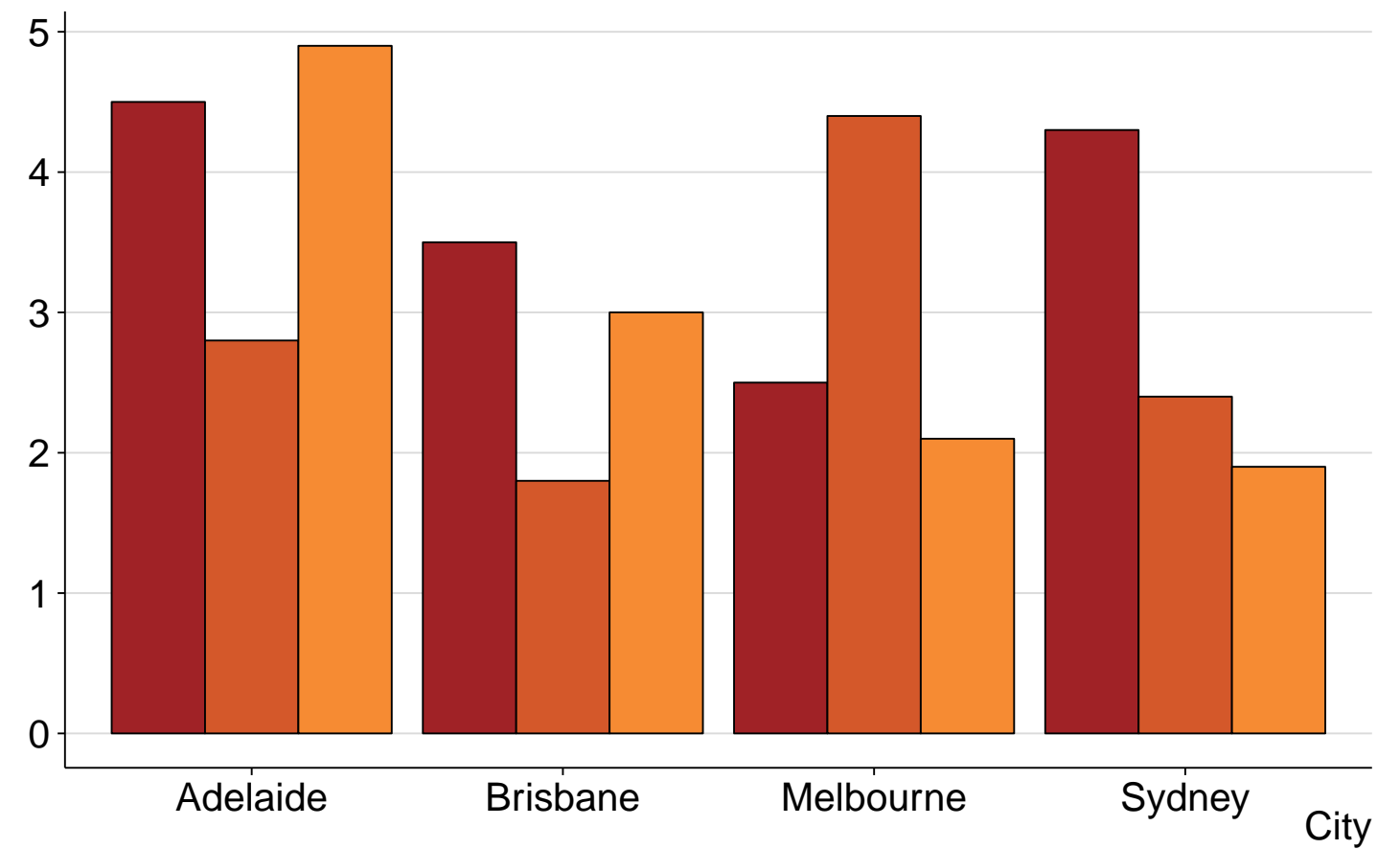
  # Alternatively, we can use geom_text to place text
  geom_text(aes(x = text.x, y = text.y,
               label = text.label, color = factor(Year)),
            size = 23/(14/5),
            hjust = 1,
            fontface = "bold")

## Warning: Removed 9 rows containing missing values (geom_text).
```



### 0.1.6 “Dodged” bar charts

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%
  # First we need to 'tidy' our data from
  # 'wide' (which it is now) to 'long'.
  # ggplot (and hence grplot) requires every
  # row to refer to a single data point ---
  # i.e. a single bar
  tidyr::gather(Year, value, -City) %>%
  # Year needs to be factor(Year) so that the colours
  # are distinctive -- leaving it makes the colors be
  # elements on a range 1994-1996
  grplot(aes(x = City, y = value, fill = factor(Year))) +
  # set stat = "identity" so that the height of the bar
  # represents values, not counts of entries
  geom_bar(stat = "identity", width = 0.9,
           # position = "dodge"
           position = "dodge")
```

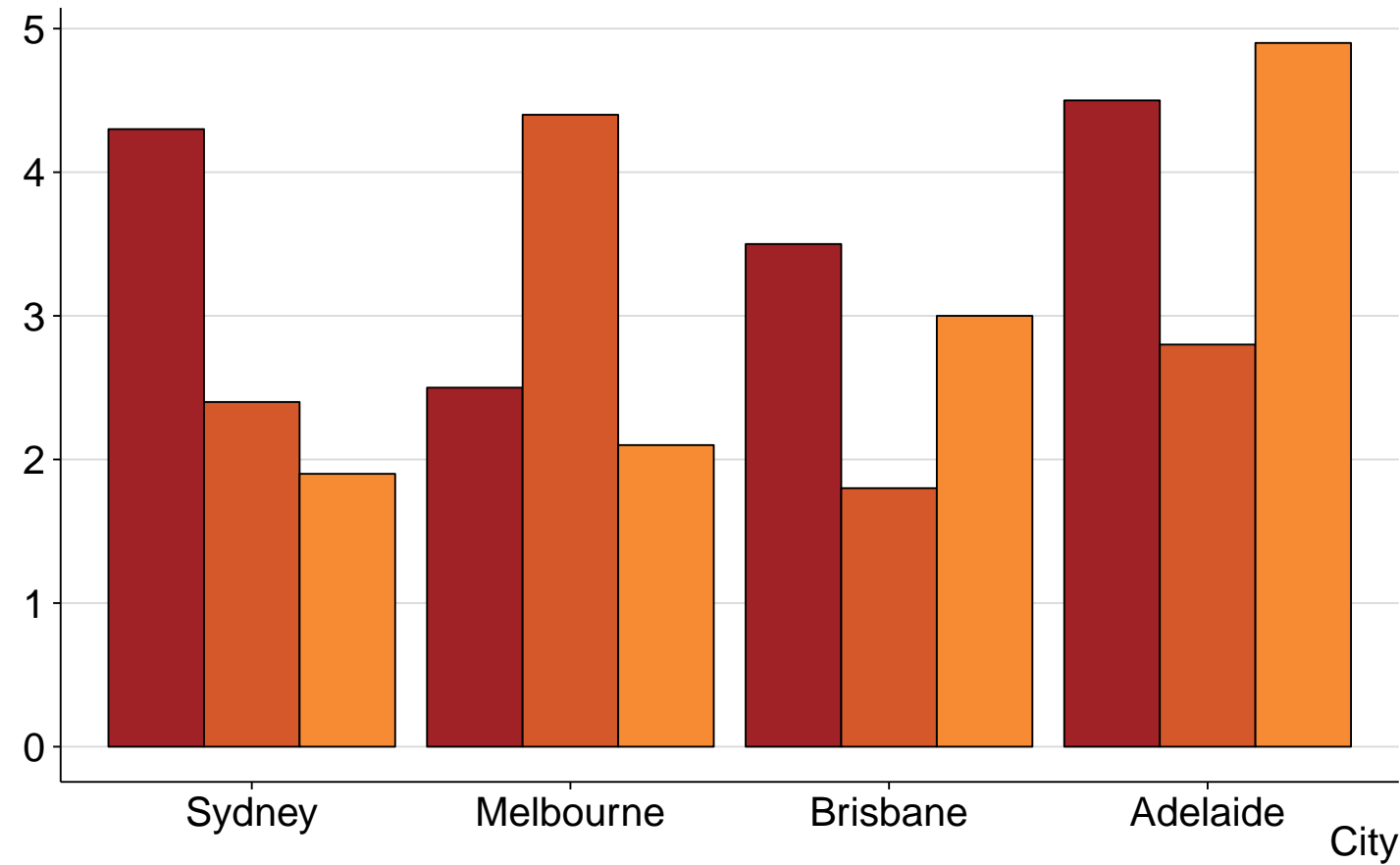


0.1.7 Order bars

```
read.table(text ="
City 1994 1995 1996
Sydney 4.3 2.4 1.9
Melbourne 2.5 4.4 2.1
Brisbane 3.5 1.8 3
Adelaide 4.5 2.8 4.9
", header = TRUE,
check.names = FALSE) %>%

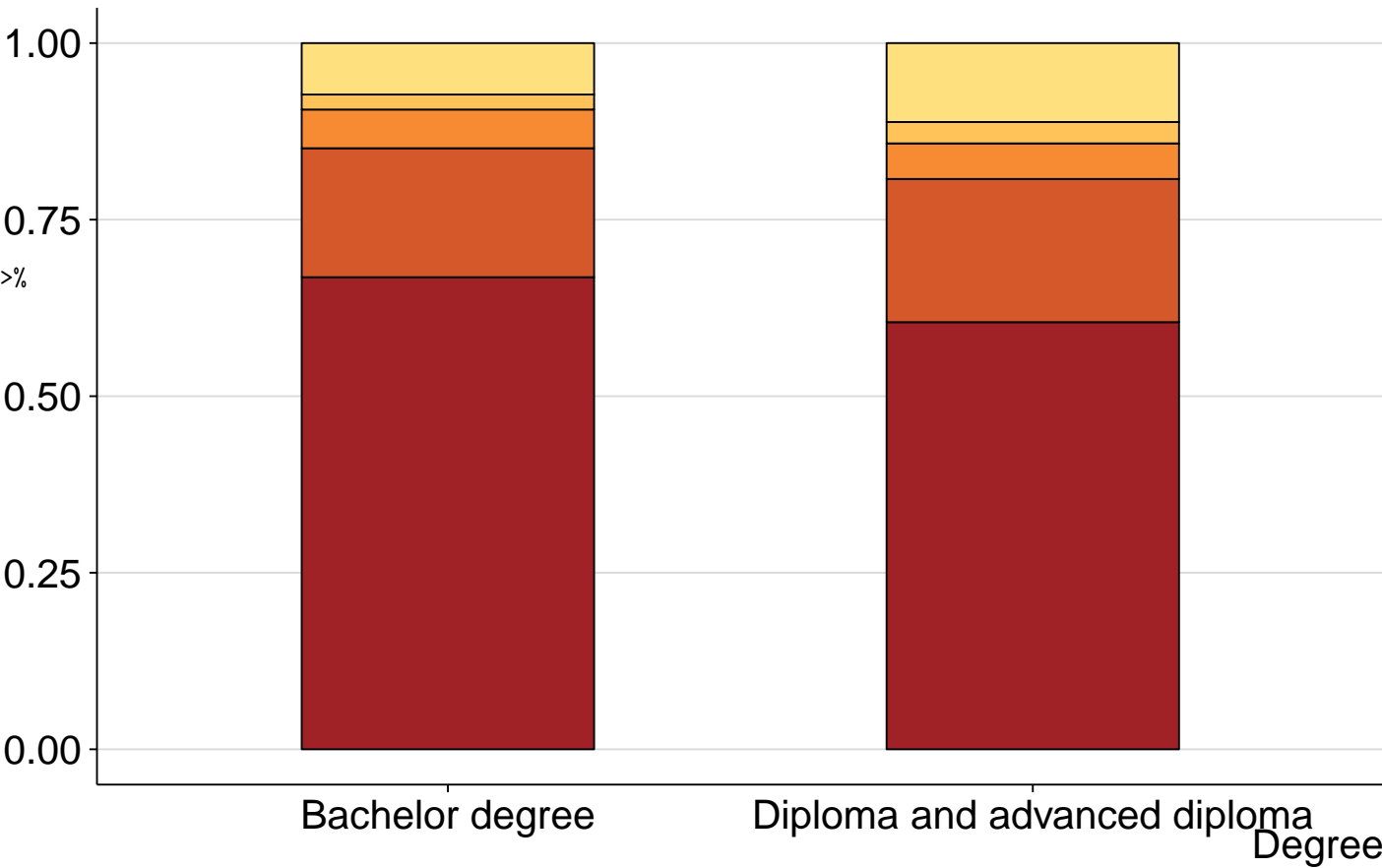
tidyr::gather(Year, value, -City) %>%
#
#
#
# use levels IMMEDIATELY before grplot to reorder bars.
mutate(City = factor(City,
                     levels = c("Sydney",
                                "Melbourne",
                                "Brisbane",
                                "Adelaide")))) %>%

grplot(aes(x = City, y = value, fill = factor(Year))) +
# set stat = "identity" so that the height of the bar
# represents values, not counts of entries
geom_bar(stat = "identity", width = 0.9,
         # position = "dodge"
         position = "dodge")
```



0.1.8 Stacked (filled) bar charts

```
read.table(text = "
Degree  Full-time Part-time Away from work Unemployed Not in LF
Diploma and advanced diploma 60.48452522 20.27784088 5.035534013 3.029462697 11.17263719
Bachelor degree 66.85055608 18.26301611 5.480946866 2.121800946 7.283680001
", header = TRUE, sep = "\t") %>%
  tidyr::gather(Status, value, -Degree) %>%
  mutate(Degree = factor(Degree, levels = c("Bachelor degree", "dummy", "Diploma and advanced diploma"))) %>%
  grplot(aes(x = Degree,
             y = value, fill = Status)) +
  geom_bar(stat = "identity", position = "fill", width = 0.5)
```

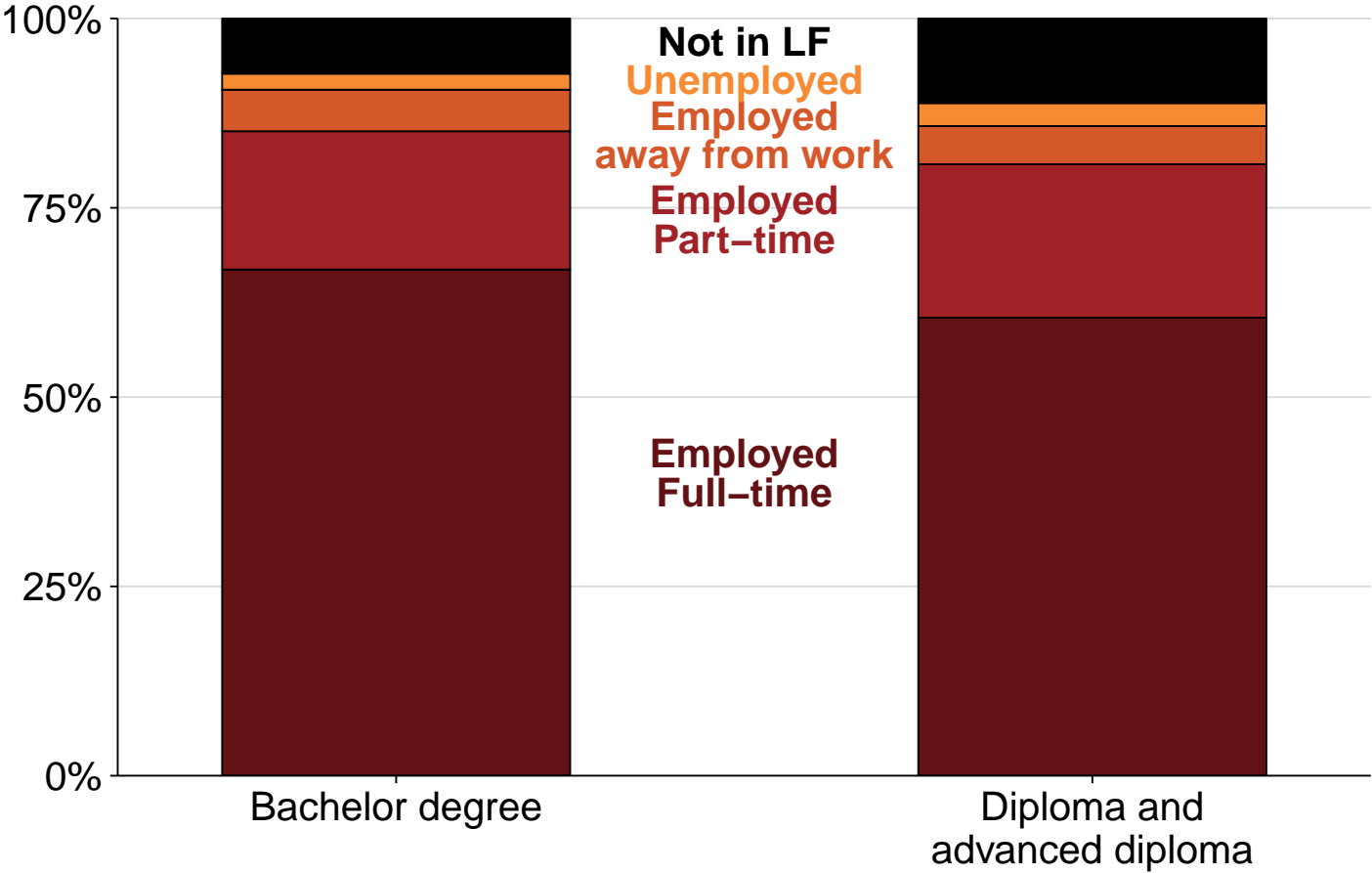


0.1.9 New colors

```
read.table(text = "
Degree    Full-time Part-time Away from work Unemployed Not in LF
Diploma and advanced diploma 60.48452522 20.27784088 5.035534013 3.029462697 11.17263719
Bachelor degree 66.85055608 18.26301611 5.480946866 2.121800946 7.283680001
", header = TRUE, sep = "\t") %>%
  tidyr::gather(Status, value, -Degree) %>%
  mutate(Degree = factor(Degree,
                        levels = c("Bachelor degree",
                                   "dummy",
                                   "Diploma and advanced diploma"),
                        labels = c("Bachelor degree",
                                   "dummy",
                                   "Diploma and\nadvanced diploma"))) %>%

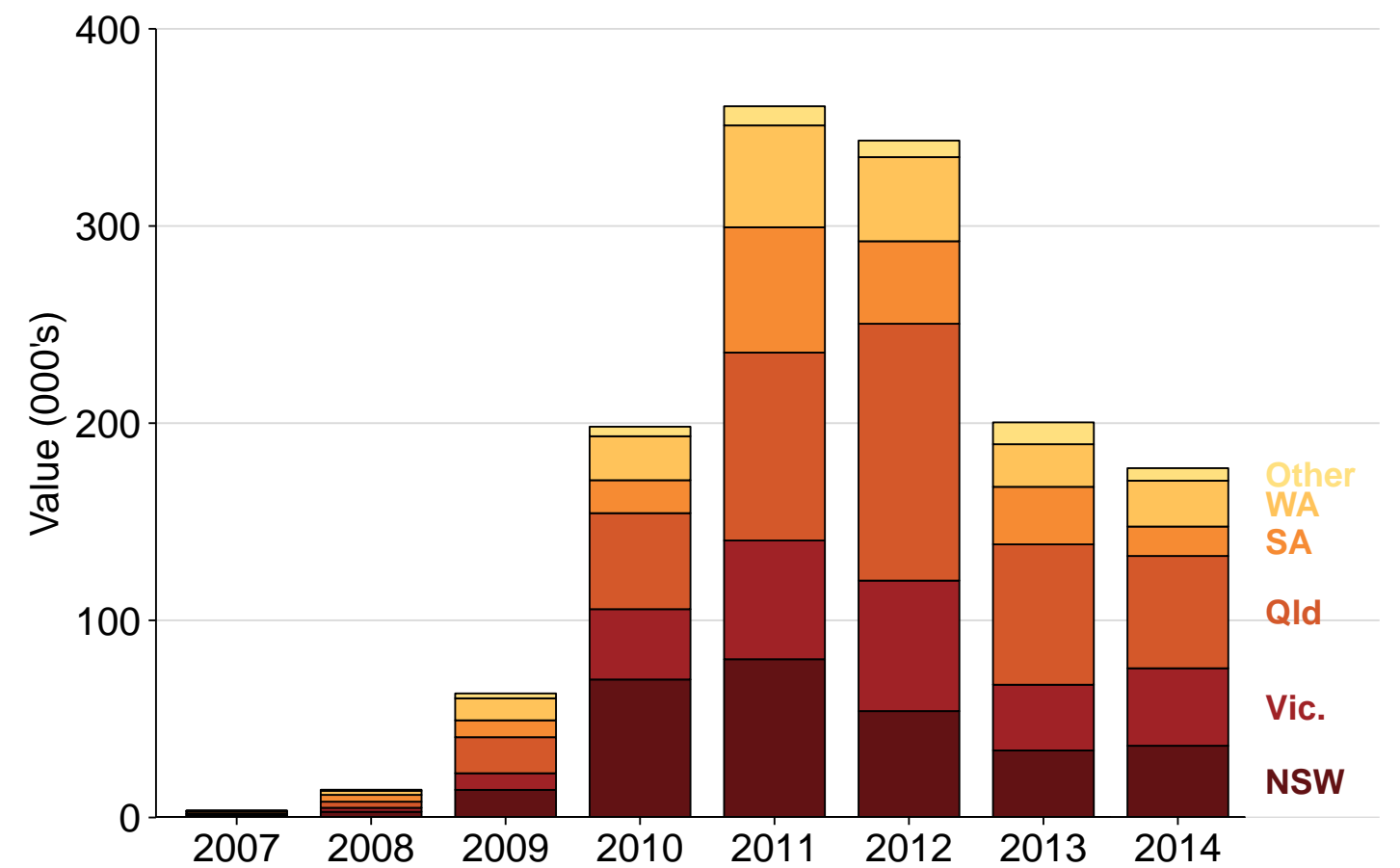
  grplot(aes(x = Degree,
            y = value, fill = Status)) +
  geom_bar(stat = "identity", position = "fill", width = 0.5) +
  scale_fill_manual(values = c(gpal(6)[1:4], "black")) +
  scale_x_discrete(expand = c(0.10,0.075)) +
  # get rid of the x-axis title (and the space allocated thereto)
  theme(axis.title = element_blank()) +
  # make data flush with plot background
  # and the axis "%"
  scale_y_continuous(expand = c(0,0),
                    label = percent) +
  annotate("text",
    x = 1.5,
    # Probably best to manually position text here!
    y = c(0.4, 0.735, 0.8465, 0.92, 0.97),
    color = c(gpal(6)[1:4], "black"),
    fontface = "bold",
    size = 23/(14/5),
    # Note use of \n to signify a new line
    label = c("Employed\nFull-time",
              "Employed\nPart-time",
              "Employed\naway from work",
              "Unemployed",
              "Not in LF"),
    lineheight = 0.8)

## Scale for 'fill' is already present. Adding another scale for 'fill',
## which will replace the existing scale.
```



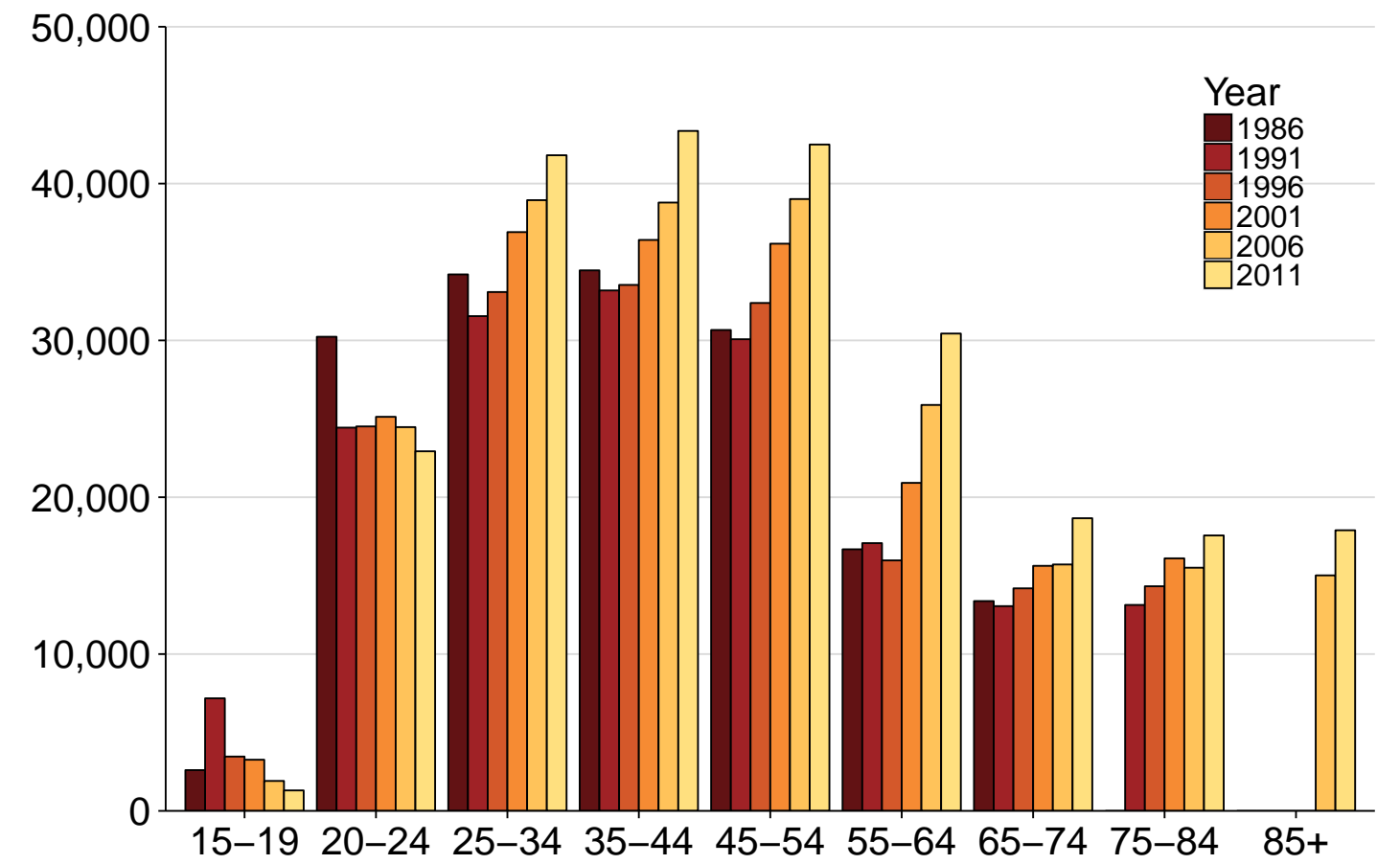
## 0.2 Labels so to appear as outside the chart (clip $x$ -axis)

```
read.table(text="Column1 NSW VIC QLD SA WA Other
2007 779 828 475 1,037 262 99
2008 2,890 2,036 3,087 3,456 2,068 527
2009 14,008 8,429 18,283 8,569 11,157 2470
2010 69,988 35,676 48,697 16,705 22,293 4849
2011 80,272 60,214 95,303 63,553 51,667 9736
2012 53,961 66,204 130,252 41,851 42,653 8399
2013 33,998 33,332 71,197 29,187 21,600 11093
2014 36,377 39,305 56,952 14,932 23,249 6353
", header = TRUE) %>%
  tidyr::gather(State, value, -Column1) %>%
  mutate(Year = Column1,
         State = factor(State,
                        labels = c("NSW", "Vic.", "Qld", "SA", "WA", "Other"))) %>%
  mutate(value = as.numeric(gsub(",", "", value))) %>%
  # text
  mutate(text.x = max(as.numeric(factor(Year))) + 0.65,
         text.label = ifelse(Year == max(Year), as.character(State), NA_character_)) %>%
  group_by(Year) %>%
  mutate(text.y = value/2 + lag(cumsum(value), default = 0)) %>%
  {
    grplot(.) +
      geom_bar(aes(x = factor(Year), fill = State, y = value),
               stat = "identity", position = "stack", width = 0.75) +
      scale_y_continuous("Value (000's)", expand = c(0,0),
                         limits = c(0, 400e3),
                         labels = function(x)comma(x/1e3)) +
      geom_text(aes(x = text.x, y = text.y, label = text.label, color = State),
                hjust = 0,
                # reduced size (from 23)
                size = 20/(14/5),
                fontface = "bold") +
      annotate("blank", x = max(as.numeric(factor(.$Year))) + 1.5, y = mean(.$value)) +
      theme(axis.title.x = element_blank(),
            axis.title.y = element_text(angle = 90,
                                         # set margin thus to prevent collision with
                                         # axis ticks
                                         margin = margin(1,10,1,1)),
            # The next steps make the axis line end at the data,
            # before the labels
            axis.line.x = element_blank()) +
      annotate("segment", x = -Inf, xend = as.numeric(factor(.$Year)) + 0.5,
               y = 0, yend = 0,
               color = "black")
  }
```



## 0.2.1

```
read.table(text="Column1 1986 1991 1996 2001 2006 2011
15-19 2600 7179 3464 3263 1910 1306
20-24 30232 24438 24515 25128 24471 22930
25-34 34207 31548 33085 36905 38943 41806
35-44 34466 33185 33538 36406 38789 43356
45-54 30664 30075 32387 36166 39013 42489
55-64 16680 17073 15973 20916 25884 30437
65-74 13378 13057 14190 15625 15712 18662
75-84 0 13132 14324 16103 15505 17565
85+ 0 0 0 0 15017 17894
", header = TRUE, check.names = FALSE) %>%
  tidyr::gather(Year, value, -Column1) %>%
  grplot(aes(x = Column1, y = value, fill = Year)) +
  geom_bar(stat = "identity", position = "dodge", width = 0.9) +
  theme(legend.position = c(0.9, 0.8),
        # x-axis clear by context
        axis.title.x = element_blank()) +
  scale_y_continuous(expand = c(0,0), limits = c(0, 50e3), label = comma)
```



### 0.3 Rowwise charts (flipped bar charts)

```
read.table(text="Intervention  Effect size
Formative evaluation 0.9
Feedback 0.73
Formative assessment 0.55
Prior achievement 0.67
Socioeconomic status 0.57
Pre-term birth weight 0.54
Decreasing disruptive behavior 0.34
Class size 0.21
", sep = "\t", header=TRUE) %>%
  mutate(is_Proposed_intervention = Intervention %in% c("Formative evaluation",
                                                         "Feedback",
                                                         "Formative assessment"),

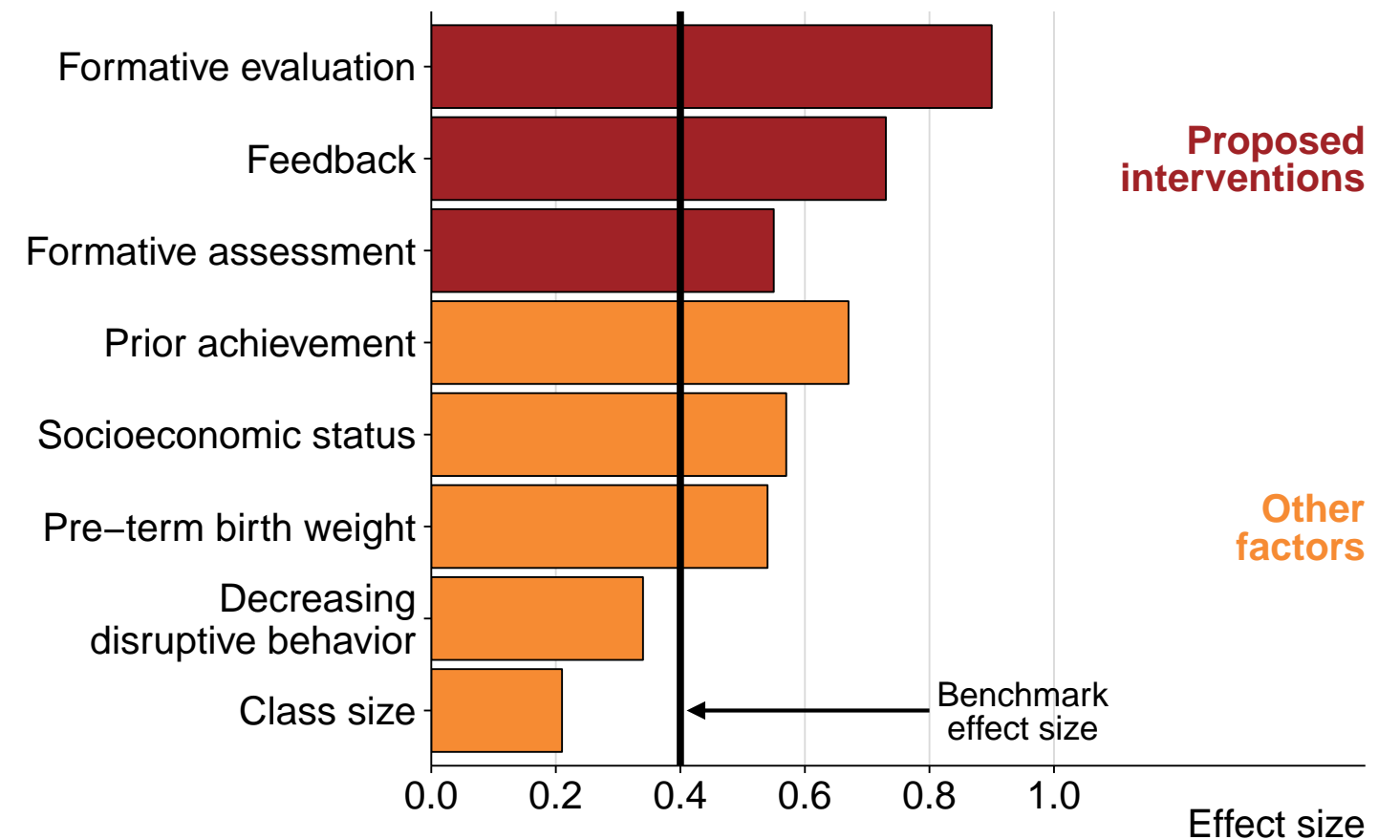
         Intervention = factor(Intervention,
                               levels = .$Intervention,
                               # if the label is too long, replace the first space
                               # with a newline
                               labels = ifelse(nchar(as.character(Intervention)) > 25,
                                                sub("\\s", "\n", as.character(Intervention)),
                                                as.character(Intervention))) %>%

  # The following says "plot in reverse of current order"
  mutate(Intervention = factor(Intervention, levels = rev(.$Intervention))) %>%

  # note `Effect size` is now `Effect.size`
  grplot(aes(x = Intervention, y = Effect.size)) +
  geom_bar(stat = "identity", width = 0.9, aes(fill = is_Proposed_intervention)) +
  # dark palette
  scale_fill_manual(values = rev(gpal(2, dark = TRUE))) +
  scale_y_continuous("Effect size",
                     expand = c(0,0),
                     limits = c(0,1.5),
                     breaks = seq(0, 1, by = 0.2)) +

  coord_flip() +
  # grid lines go the wrong way!
  theme(panel.grid.major.y = element_blank(),
        panel.grid.major.x = element_line()) +
  annotate("text", x = c(7, 3), y = 1.5,
          label = c("Proposed\ninterventions",
                    "Other\nfactors"),
          color = gpal(2,T),
          hjust = 1,
          size = 23/(14/5),
          fontface = "bold",
          lineheight = 0.8) +
  geom_hline(yintercept = 0.4, size = 2) +
  # arrow
  annotate("segment", y = 0.8, yend = 0.43, x = 1, xend = 1,
          size = 1.1
          # basic arrow (bit curvy for my liking)
          arrow = arrow(length = unit(0.5, "lines"), # size of arrowhead
                        type = "closed")
  ) +
  annotate("text", x = 1, y = 0.95, size = 20/(14/5),
          lineheight = 0.8,
          label = "Benchmark\neffect size") +
  # arrowhead
  annotate("polygon", x = c(1, 0.9, 1.1), y = c(0.41, 0.44, 0.44), fill = "black")

## Scale for 'fill' is already present. Adding another scale for 'fill',
## which will replace the existing scale.
```



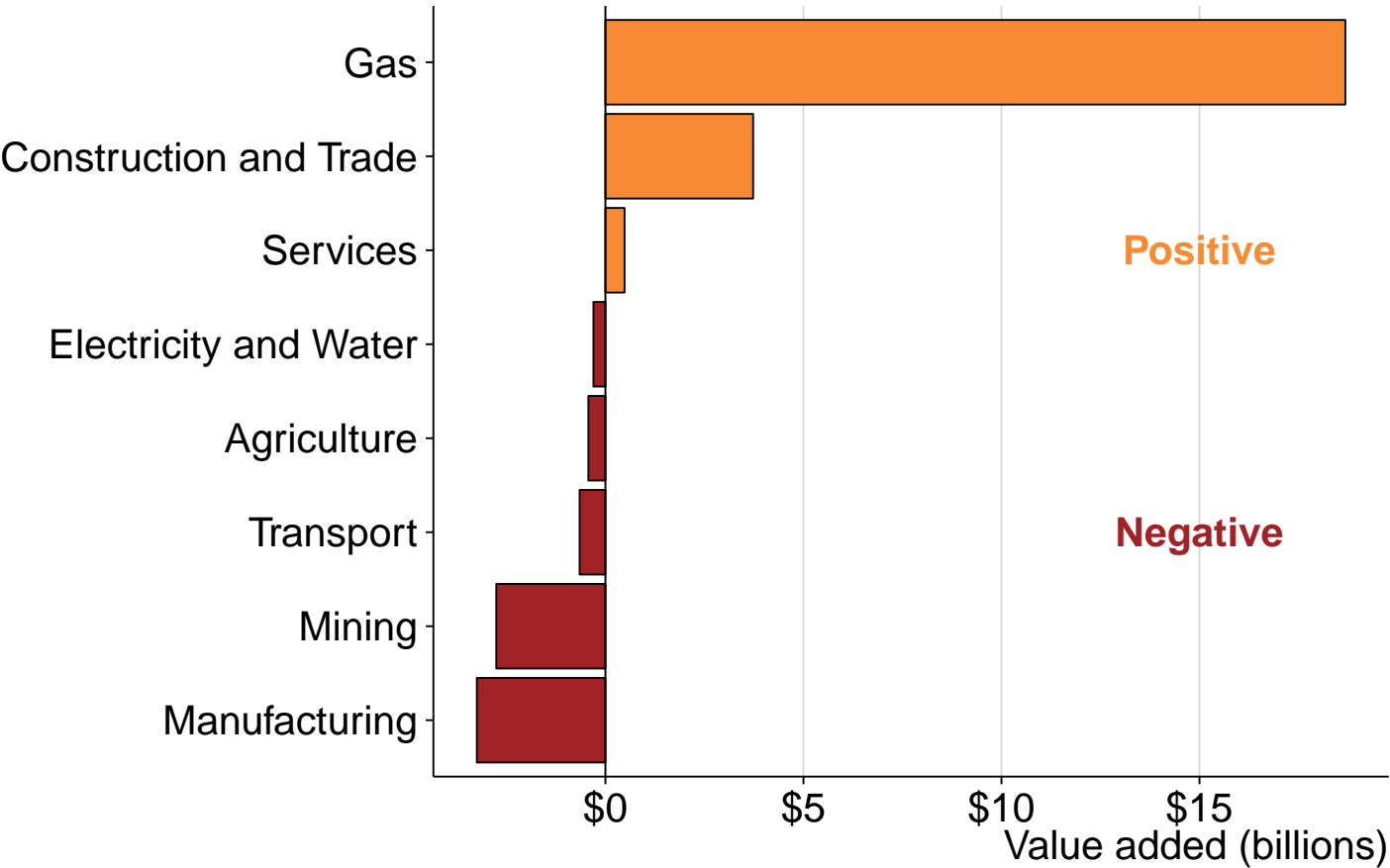


0.4 Bar chart signed colours

```
read.table(text="Sector Value Added
Gas 18.683
Construction and Trade 3.73
Services 0.483
Electricity and Water -0.302
Agriculture -0.43
Transport -0.652
Mining -2.756
Manufacturing -3.247
", header = TRUE, sep = "\t") %>%
  arrange(desc(Value.Added)) %>% # redundant here
  mutate(Sector = factor(Sector, levels = rev(.$Sector))) %>%
  grplot(aes(x = Sector, y = Value.Added)) +
  geom_bar(stat = "identity", width = 0.9, aes(fill = Value.Added > 0)) +
  scale_fill_manual(values = gpal(2, T)) +
  scale_y_continuous("Value added (billions)",
    label = grattan_dollar) +

  coord_flip() +
  annotate("text",
    x = c(3, 6),
    y = 15,
    label = c("Negative", "Positive"),
    size = 23/(14/5),
    fontface = "bold",
    color = gpal(2, T)) +
  theme(panel.grid.major.x = element_line(),
    panel.grid.major.y = element_blank(),
    #
    axis.title.x = element_text(margin = margin(5,1,1,1))) +
  geom_hline(yintercept = 0, color = "black")

## Scale for 'fill' is already present. Adding another scale for 'fill',
## which will replace the existing scale.
## Warning: Stacking not well defined when ymin != 0
```



0.5 Bar charts faceted

```
read.table(text = "Reliability_level Solar PV Battery Grid
95 per cent 8,200 26,000 0
99 per cent 11,200 41,000 0
99.9 per cent 16,200 56,000 0
Grid-connected 0 0 12,910
", sep = "\t", header = TRUE) %>%
  tidyr::gather(Electricity_source, value, -Reliability_level) %>%
  mutate(value = as.numeric(gsub(",", "", value)),
         is_grid_connected = factor(ifelse(grepl("Grid.connected", Reliability_level),
                                           "Grid-connected",
                                           "Off-grid reliability level"),
                                   levels = c("Off-grid reliability level",
                                              "Grid-connected"))) %>%

  group_by(Reliability_level) %>%
  mutate(text.middle.y = value/2 + cumsum(lag(value, default = 0)),
         text.top.y = ifelse(Electricity_source == first(Electricity_source),
                             max(cumsum(value)),
                             NA_real_)) %>%

  grplot(aes(x = Reliability_level, y = value, fill = Electricity_source)) +
  geom_bar(stat = "identity", position = "stack", width = 0.9) +
  scale_fill_manual(values = rev(gpal(3))) +
  geom_text(aes(y = text.middle.y,
               label = ifelse(value > 0,
                              grattan_dollar(value),
                              NA_real_)),
            size = 23/(14/5)) +
  geom_text(aes(y = text.top.y, label = grattan_dollar(text.top.y)),
            vjust = -0.25,
            size = 23/(14/5)) +
  # with all the bars marked, no need for y-axis
  theme(axis.text.y = element_blank(),
        axis.line.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title.x = element_blank(),
        panel.grid.major = element_blank(),
        axis.text.x = element_text(margin = margin(1, 1, 5.5, 1))) +
  scale_y_continuous(expand = c(0,0), limits = c(0, 80e3)) +
  facet_grid(~is_grid_connected, scales = "free_x", space = "free", switch = "x") +
  theme(strip.background = element_rect(fill = grey(0.5)))

## Warning: attributes are not identical across measure variables; they will be dropped
## Scale for 'fill' is already present. Adding another scale for 'fill',
## which will replace the existing scale.
## Warning: Removed 5 rows containing missing values (geom_text).
## Warning: Removed 8 rows containing missing values (geom_text).
```

