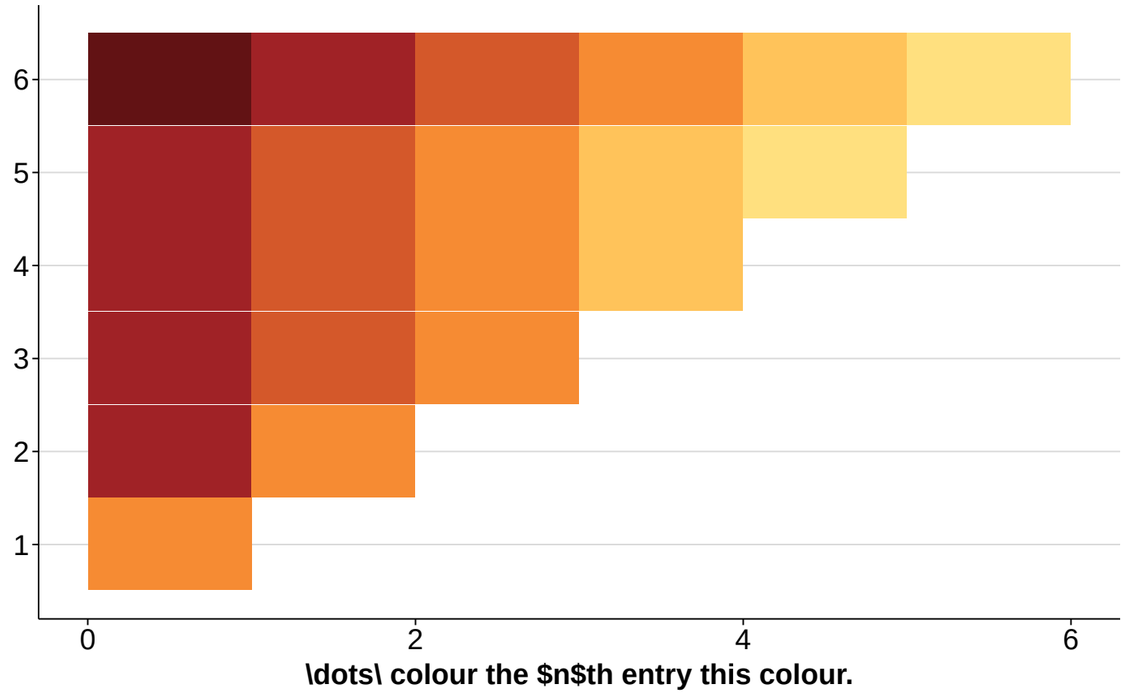
```
library(knitr)
knitr::opts_chunk$set(fig.show = "hide",
                      fig.path = 'atlas/',
                      fig.width = 11,
                      fig.height = 7,
                      tidy = FALSE,
                      message = FALSE,
                      out.width = "11in",
    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

# Nicer tildes
hook_source = knit_hooks$get('source')
knit_hooks$set(source = function(x, options) {
  txt = hook_source(x, options)
  # extend the default source hook
  gsub('~', '\\\\mytilde', txt)
})
```

```
library(directlabels)
library(ggplot2)
library(scales)
library(grattan)
library(tidyr)
library(dplyr)
library(readr)
library(magrittr)
library(lubridate)
library(readxl)
library(data.table)
library(grid)
```

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

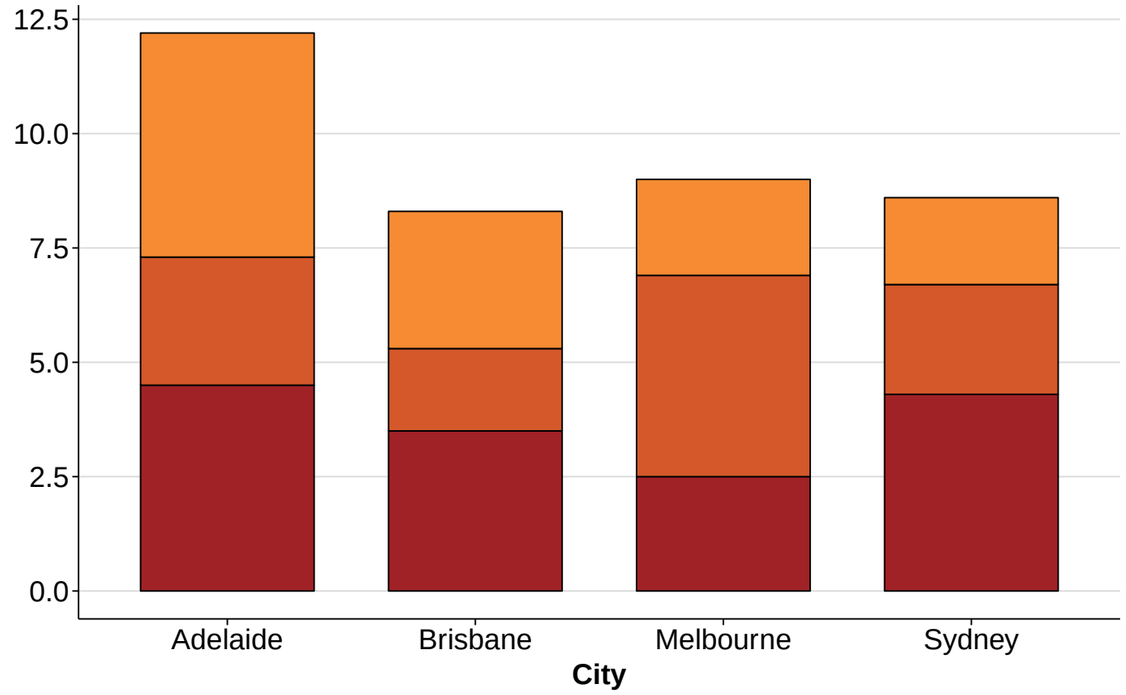
```
p <- grplot(NULL)
for (y in 1:6){
  for (x in 1:6){
    p <- p + annotate("rect",
                      xmin = x - 1, xmax = x,
                      ymin = y - 0.99, ymax = y,
                      fill = gpal(y)[x])
  }
}
p +
  scale_y_continuous("With this number of categories",
                     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
```

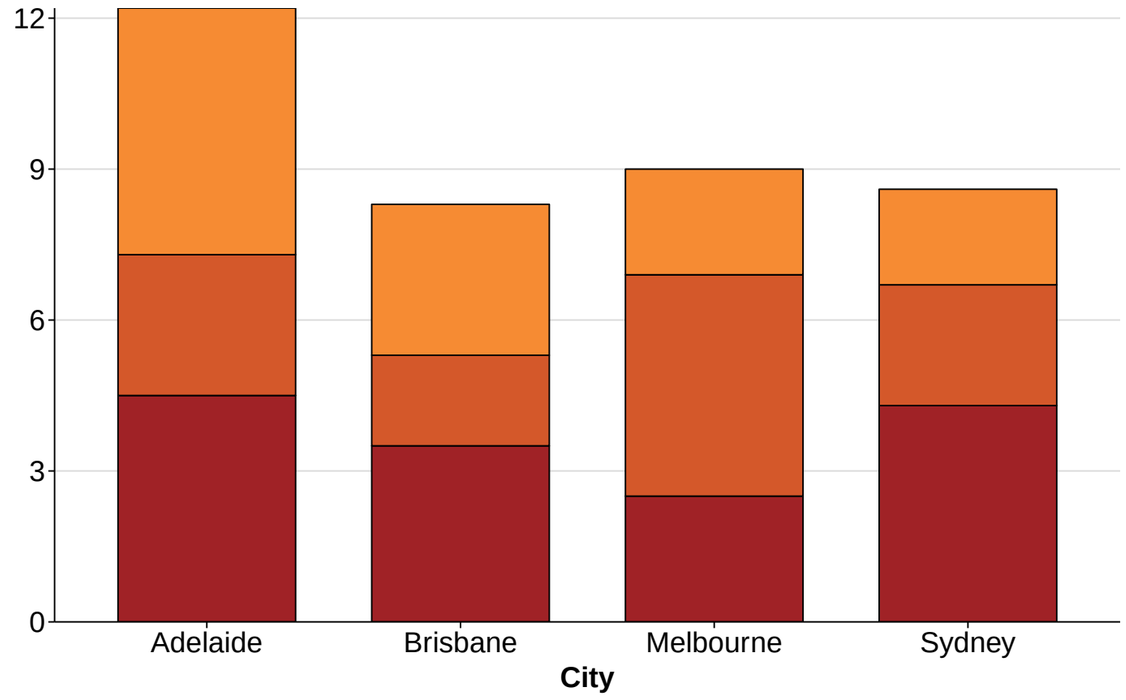
0.1 Bar charts

```
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)
```



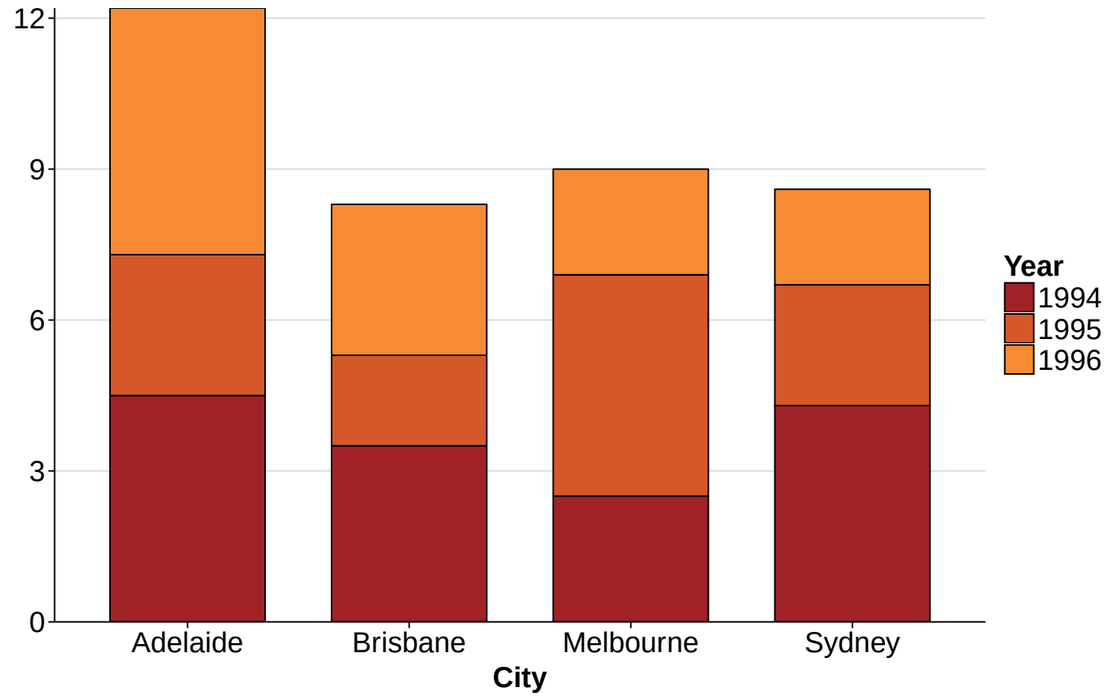
0.1.1 Axes flush with data

```
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")
```

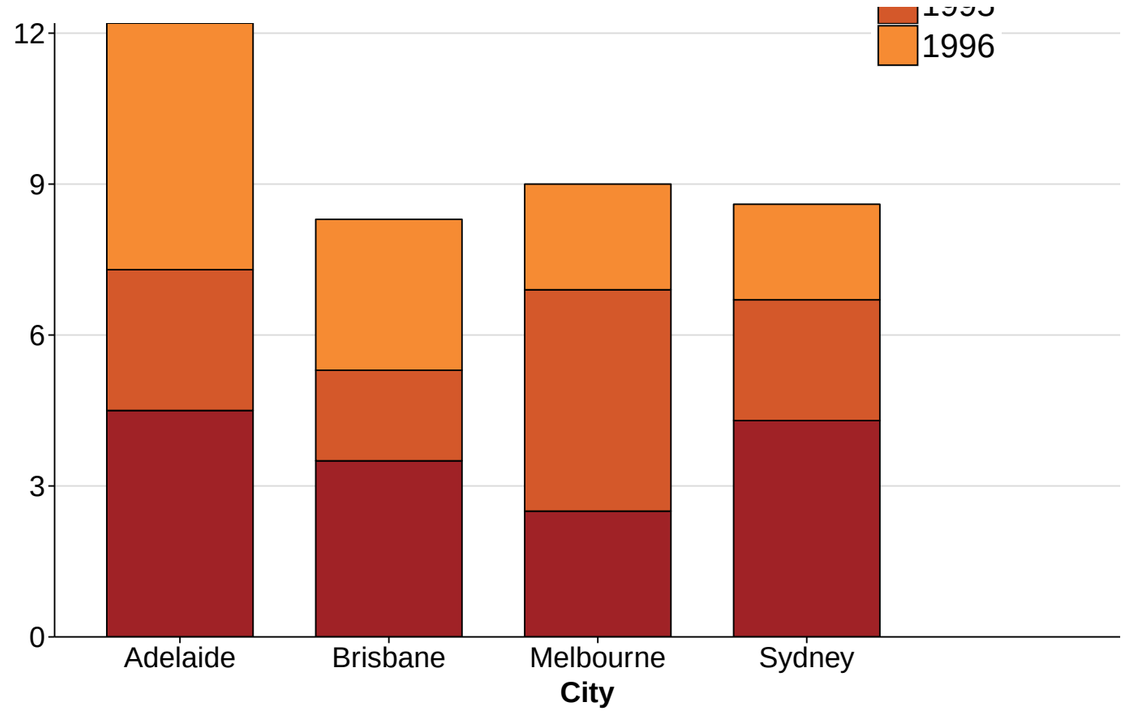


0.1.3 Add legend over plot

```
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"))
```

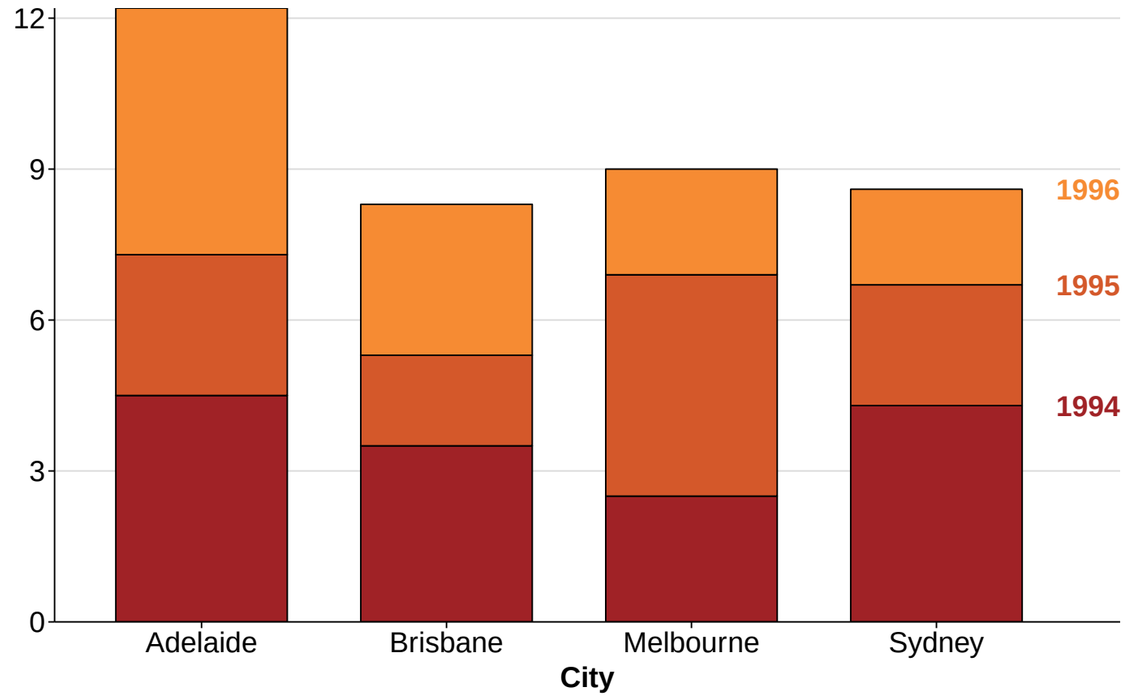
0.1.4 Adding space for legend

```
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_)
```



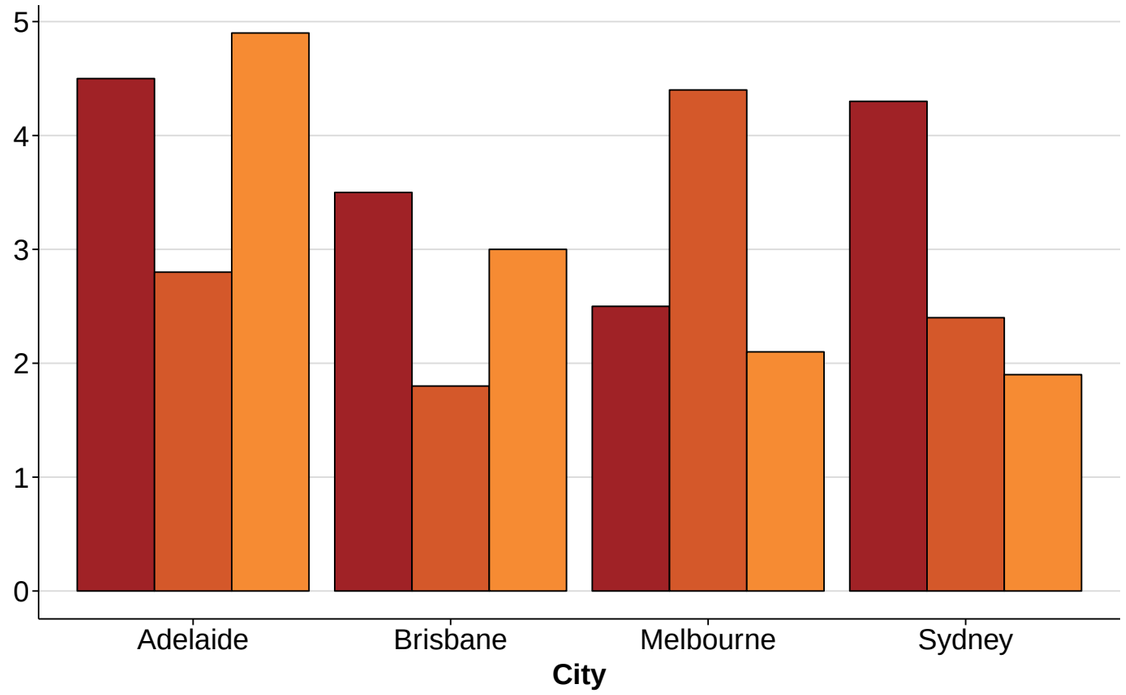
0.1.5 Coloured text as legend

```
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)),
          hjust = 1,
          fontface = "bold",
          na.rm = TRUE)
```



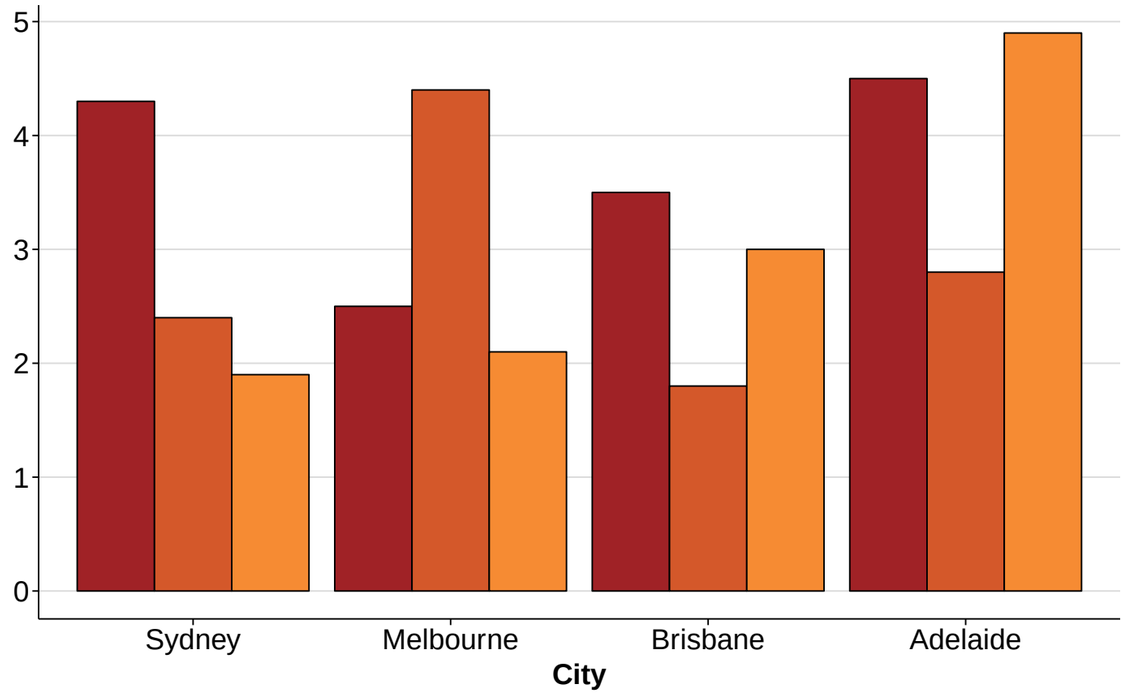
0.1.6 "Dodged" bar charts

```
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")
```



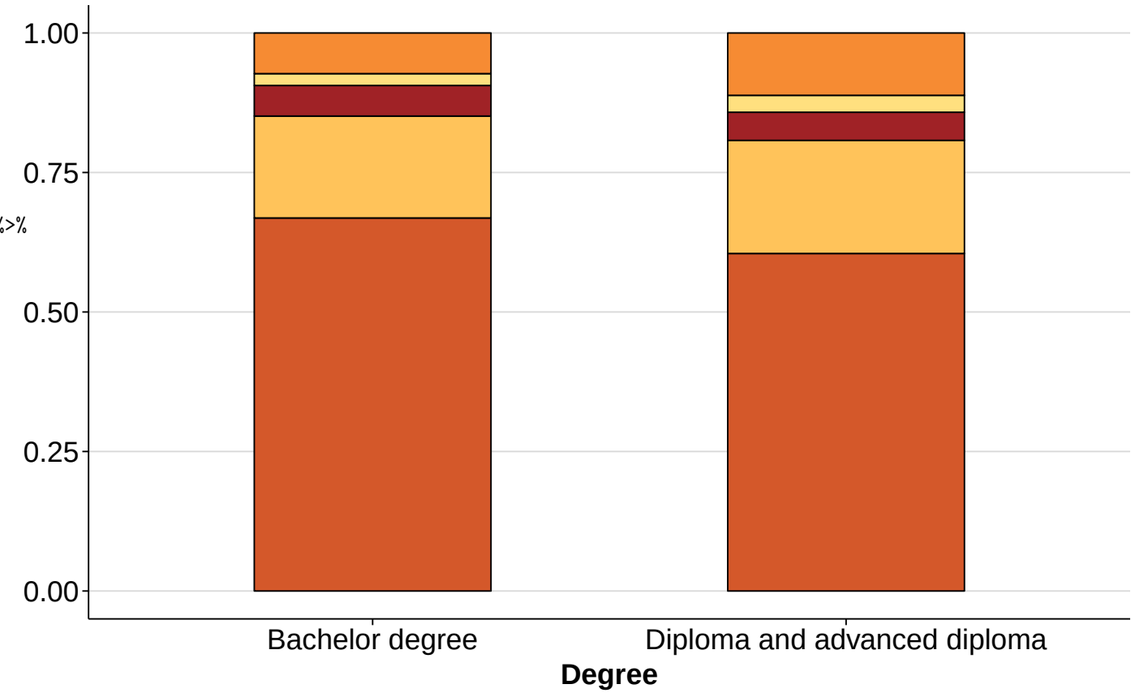
0.1.7 Order bars

```
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")
```



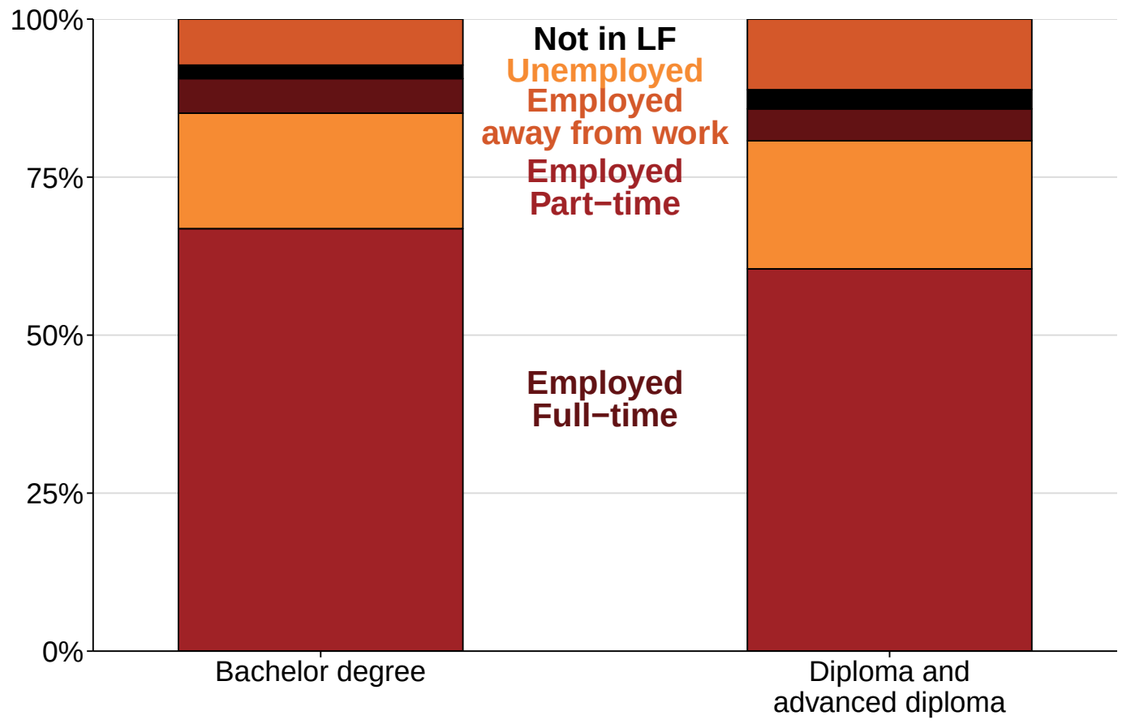
0.1.8 Stacked (filled) bar charts

```
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)
```



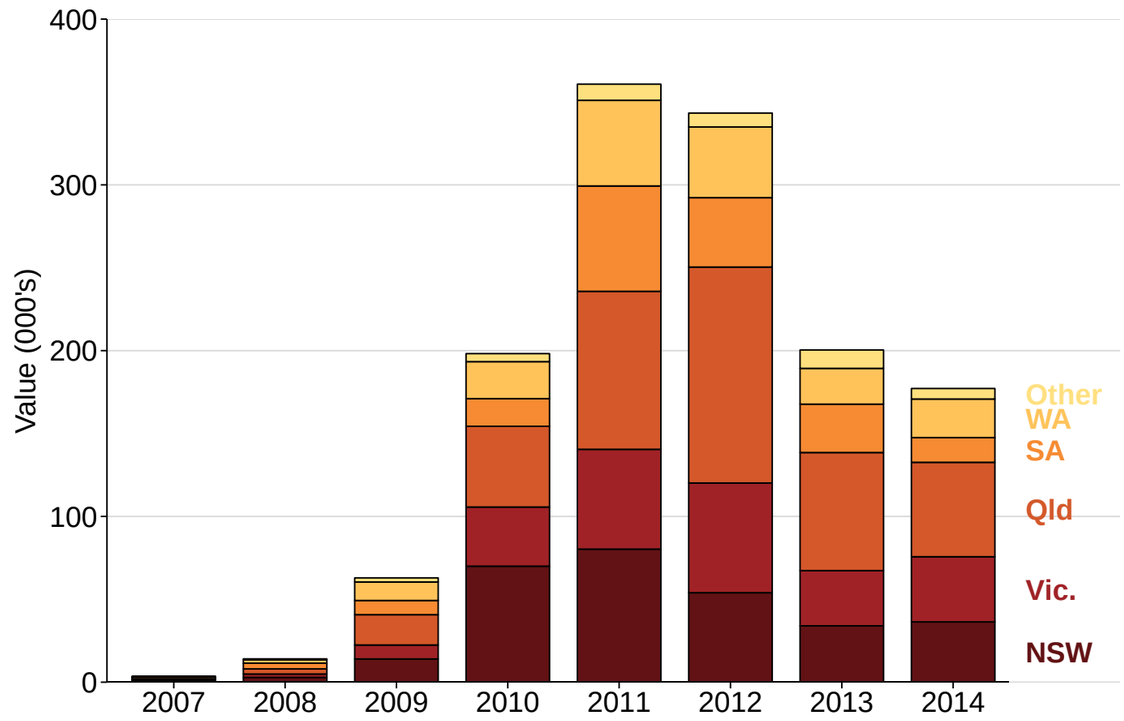
0.1.9 New colors

```
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)
```



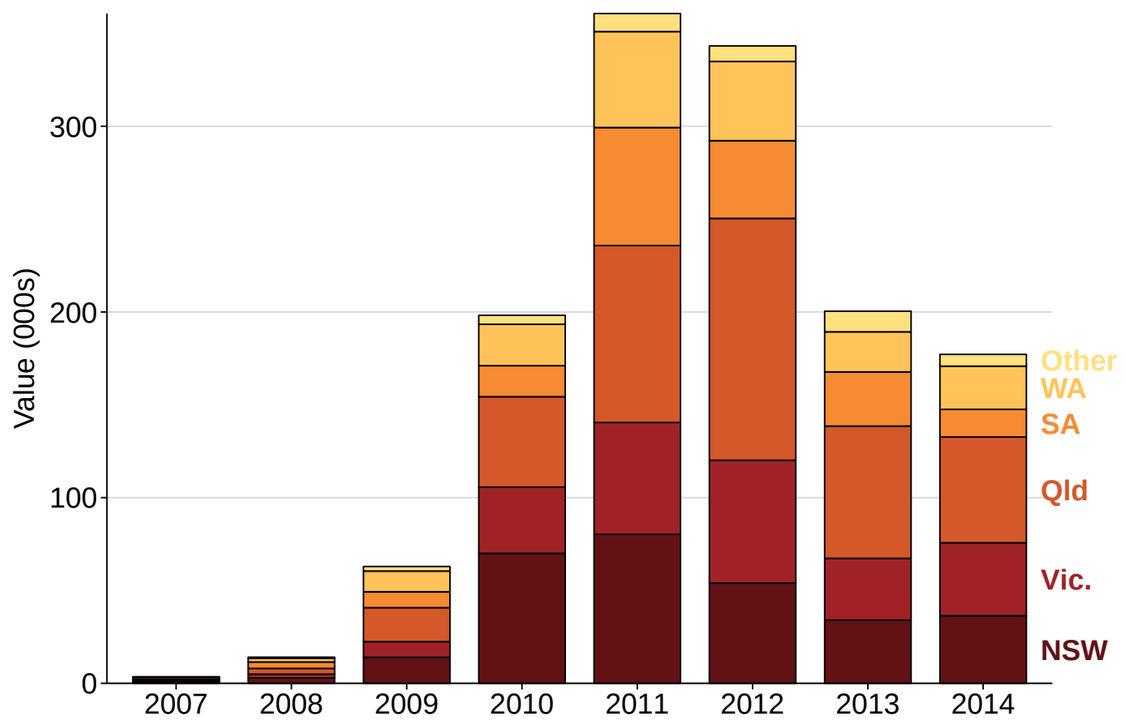
0.2 Labels so as to appear outside the chart (clip x-axis)

```
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, factor_key = TRUE) %>%
  mutate(Year = Column1,
         State = factor(State,
                        labels = c("NSW", "Vic.", "Qld", "SA", "WA", "Other"),
                        ordered = TRUE)) %>%
  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")
  }
```



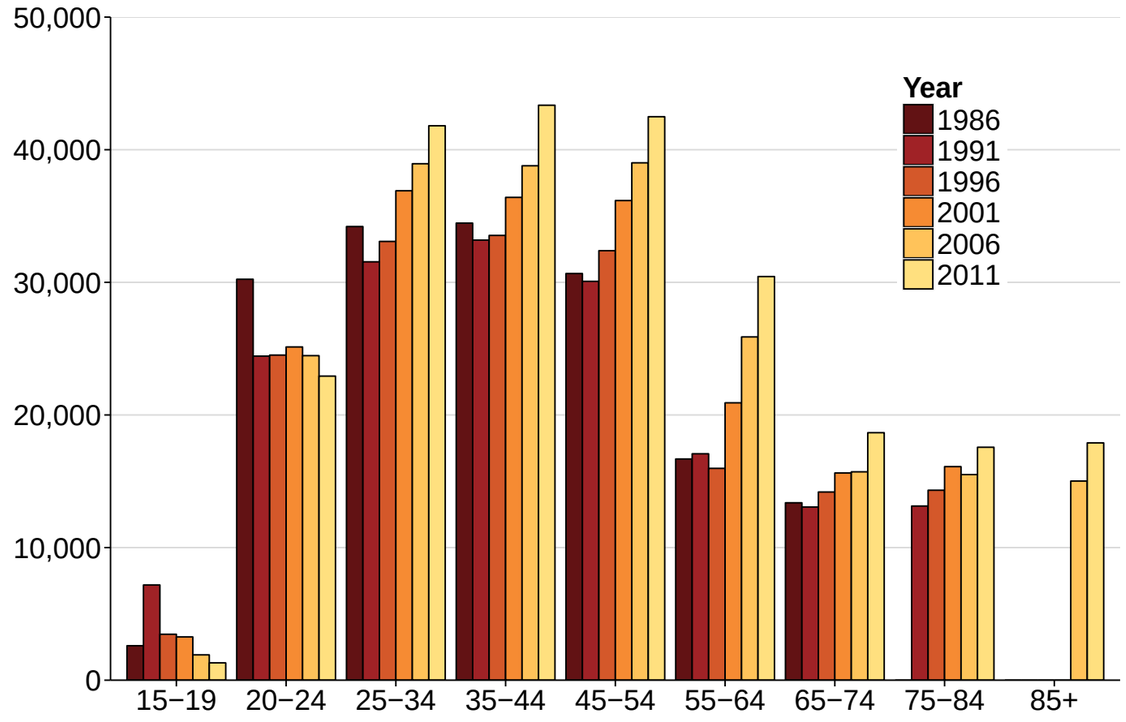
0.3 Grattan function for stacked bar chart with right labels

```
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) %>%
  mutate_each(funs(as.character), -Column1) %>%  # avoids type.convert warning
  tidyr::gather(State, value, -Column1, factor_key = TRUE) %>%
  mutate(Year = Column1,
         State = factor(State,
                        labels = c("NSW", "Vic.", "Qld", "SA", "WA", "Other"))) %>%
  mutate(y = as.numeric(gsub(",", "", value)),
         x = factor(Year, ordered = TRUE),
         fill = factor(State, ordered = TRUE)) %>%
  stacked_bar_with_right_labels(
    barwidth = 0.75,
    theme.args = list(text = element_text(size = 20),
                      axis.title.x = element_blank(),
                      axis.title.y = element_text(angle = 90, margin = margin(11, 11, 11, 11, "pt"))),
    scale_y_args = list(name = "Value (000s)",
                        label = function(x) x / 1000,
                        expand = c(0,0)))
```



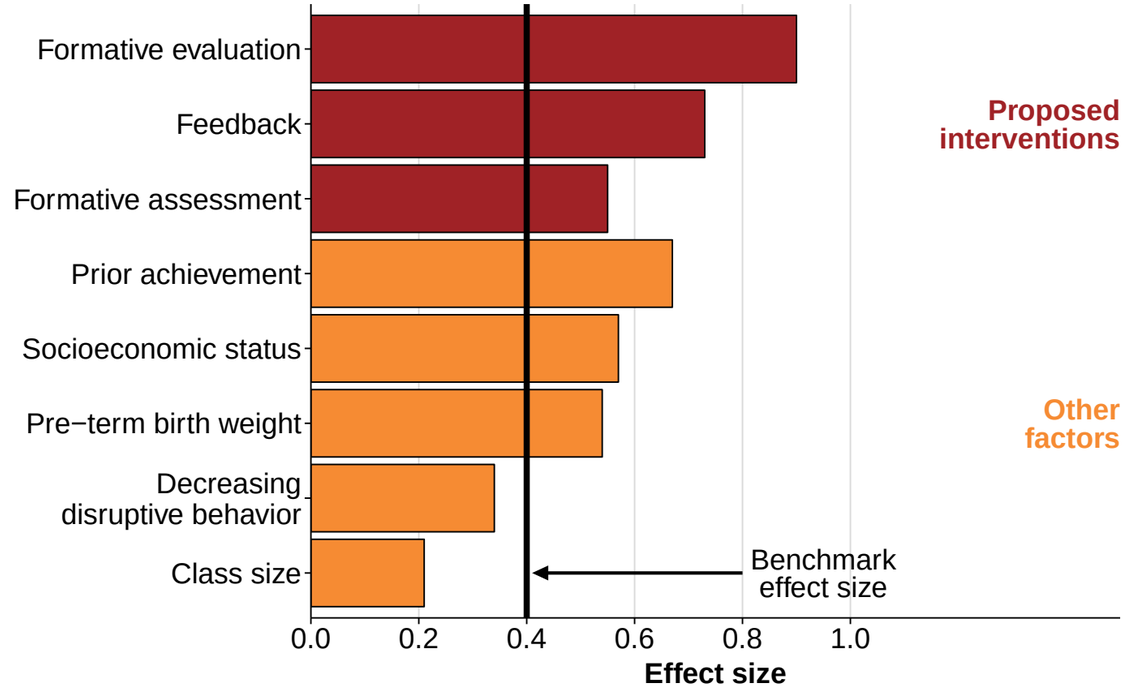
0.3.1

```
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, 0.9),
        legend.justification = c(1, 0.9),
        # x-axis clear by context
        axis.title.x = element_blank()) +
  scale_y_continuous(expand = c(0,0), limits = c(0, 50e3), label = comma)
```



0.4 Rowwise charts (flipped bar charts)

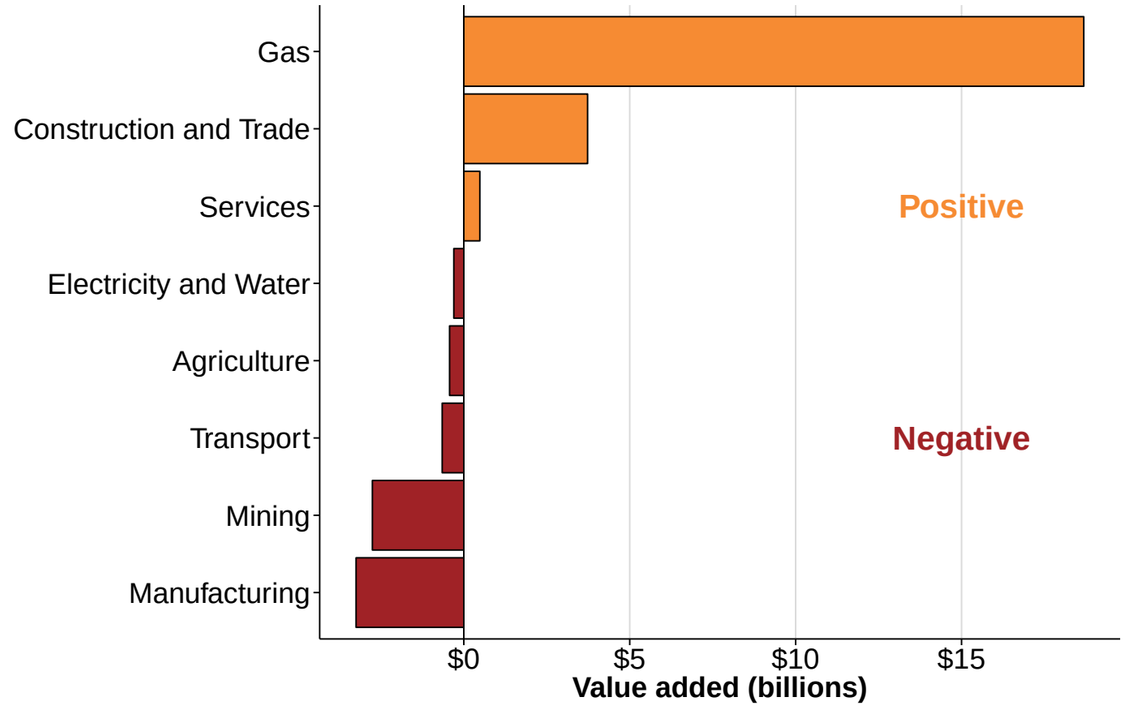
```
read.table(text="Intervention  Effect size
Formative evaluation 0.9
Feedback 0.73
Formative assessment 0.55
Prior achievement 0.67
Socioeconomic status 0.57
Pre-term birth weight 0.54
Decreasing disruptive behavior 0.34
Class size 0.21
", sep = "\t", header=TRUE) %>%
  mutate(is_Proposed_intervention = 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),
         # reverse puts the palette in the opposite direction
         reverse = TRUE) +
  geom_bar(stat = "identity", width = 0.9, aes(fill = is_Proposed_intervention)) +
  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),
           hjust = 1,
           size = 20/(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")
```



0.5 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")
```

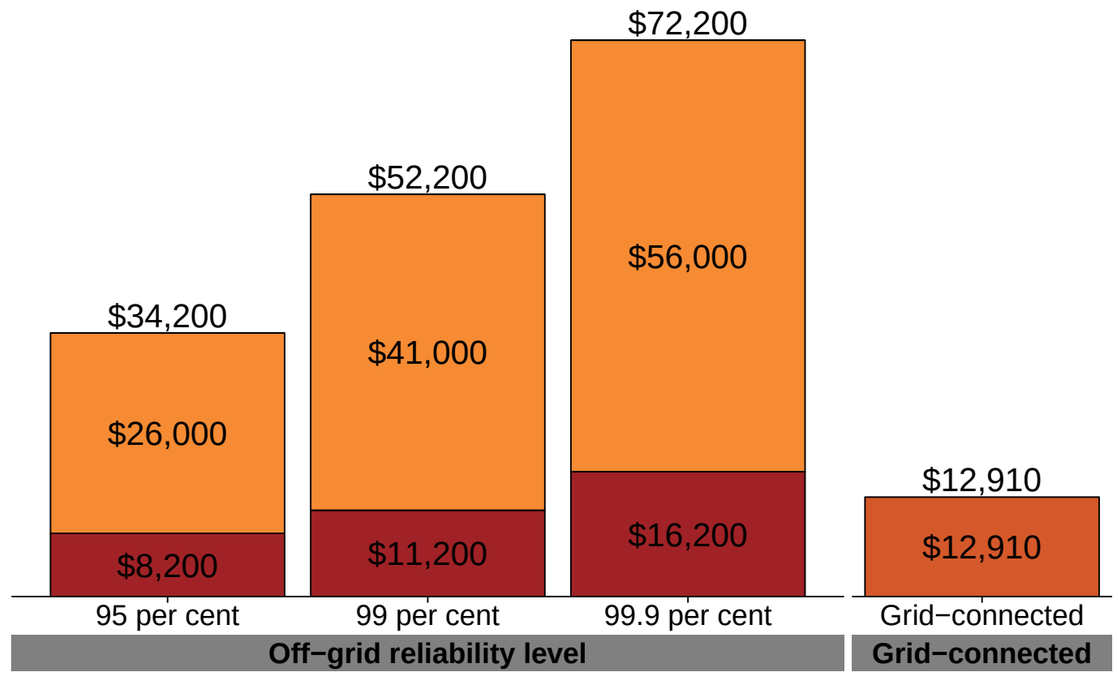
```
## Warning:  Stacking not well defined when ymin != 0
```



0.6 Bar charts facetted

```
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
## Warning:  Removed 5 rows containing missing values (geom_text).
## Warning:  Removed 8 rows containing missing values (geom_text).
```



1 Line charts

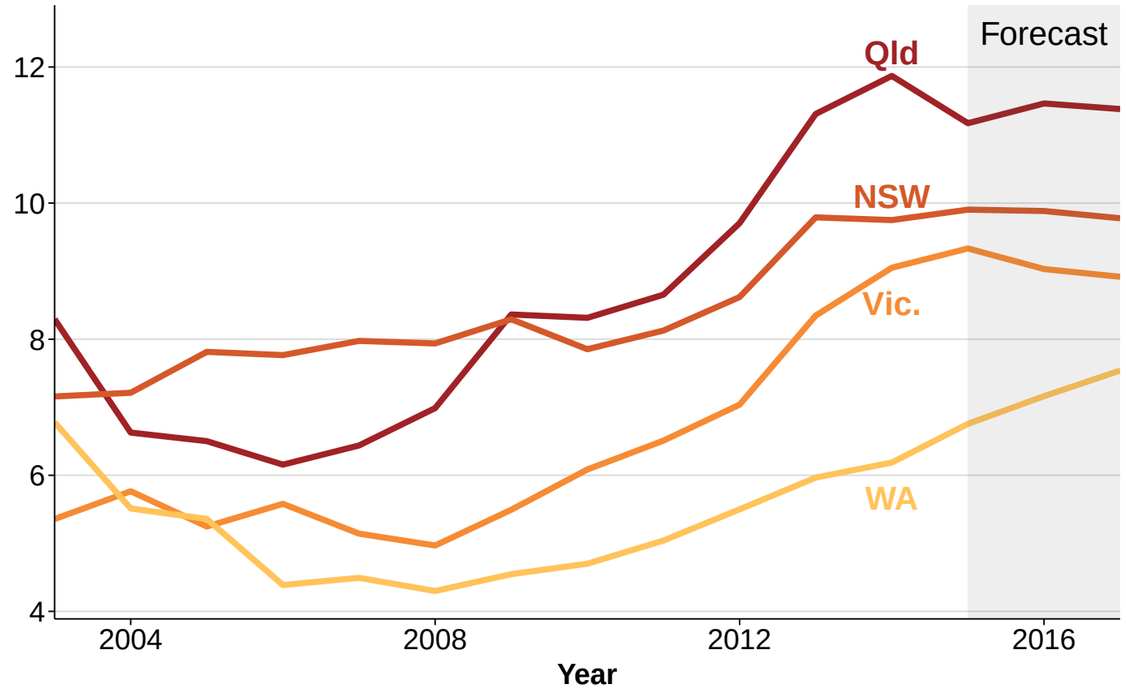
1.1 Basic time series

```
read_csv("./Supp-Data/line-chart-1.csv") %>%
  tidyr::gather(Country, value, -Month) %>%
  mutate(Date = as.Date(paste0("01-", Month), "%d-%b-%y")) %>%
  group_by(Country) %>%
  #
  # Place the labels at maxima
  mutate(text.label = ifelse(value == max(value), as.character(Country), NA_character_)) %>%
  grplot(aes(x = Date, y = value, color = Country)) +
  geom_line() +
  geom_text(aes(label = text.label),
            size = 23/(14/5),
            fontface = "bold",
            vjust = -0.35) +
  theme(axis.title = element_blank())

## Warning:  Removed 180 rows containing missing values (geom_text).
```

1.2 Line chart with forecast

```
read.table(text="Year  VIC NSW QLD WA
2003 5.356414724 7.158303299 8.295067397 6.779373035
2004 5.765232407 7.212470457 6.627270564 5.512428448
2005 5.247275775 7.813739286 6.501705246 5.357896217
2006 5.579882397 7.766102172 6.156096264 4.386993275
2007 5.142464026 7.974499497 6.437996373 4.492579222
2008 4.967862882 7.937344521 6.986955138 4.299346541
2009 5.493190785 8.292898991 8.363056636 4.546391753
2010 6.085006168 7.852222696 8.313826172 4.700406152
2011 6.509222847 8.126837463 8.654098243 5.040561095
2012 7.038135416 8.619037843 9.705296782 5.499212089
2013 8.347365139 9.790432725 11.31027881 5.965909091
2014 9.049923004 9.750211753 11.8696421 6.187004075
2015 9.333387206 9.904002678 11.17400215 6.75574023
2016 9.031231241 9.884136122 11.46456268 7.162695011
2017 8.916867397 9.77732708 11.38238887 7.53921181
", header=TRUE) %>%
  tidyr::gather(State, value, -Year) %>%
  mutate(State = factor(State, levels = c("QLD", "NSW", "VIC", "WA"),
                        labels = c("Qld", "NSW", "Vic.", "WA"))) %>%
  mutate(label.text = ifelse(Year == 2014, as.character(State), NA_character_)) %>%
  grplot(aes(x = Year, y = value, color = State)) +
  geom_line() +
  geom_text(aes(label = label.text,
                # put labels above if QLD/NSW, else below
                vjust = ifelse(State %in% c("Qld", "NSW"),
                               -0.5,
                               2.0)),
            fontface = "bold",
            size = 23/(14/5)) +
  scale_x_continuous(expand = c(0,0)) +
  annotate("rect", xmin = 2015, xmax = Inf, ymin = -Inf, ymax = Inf,
           alpha = 0.1) +
  annotate("text", x = 2016, y = 12.5, label = "Forecast",
           size = 23/(14/5))
```

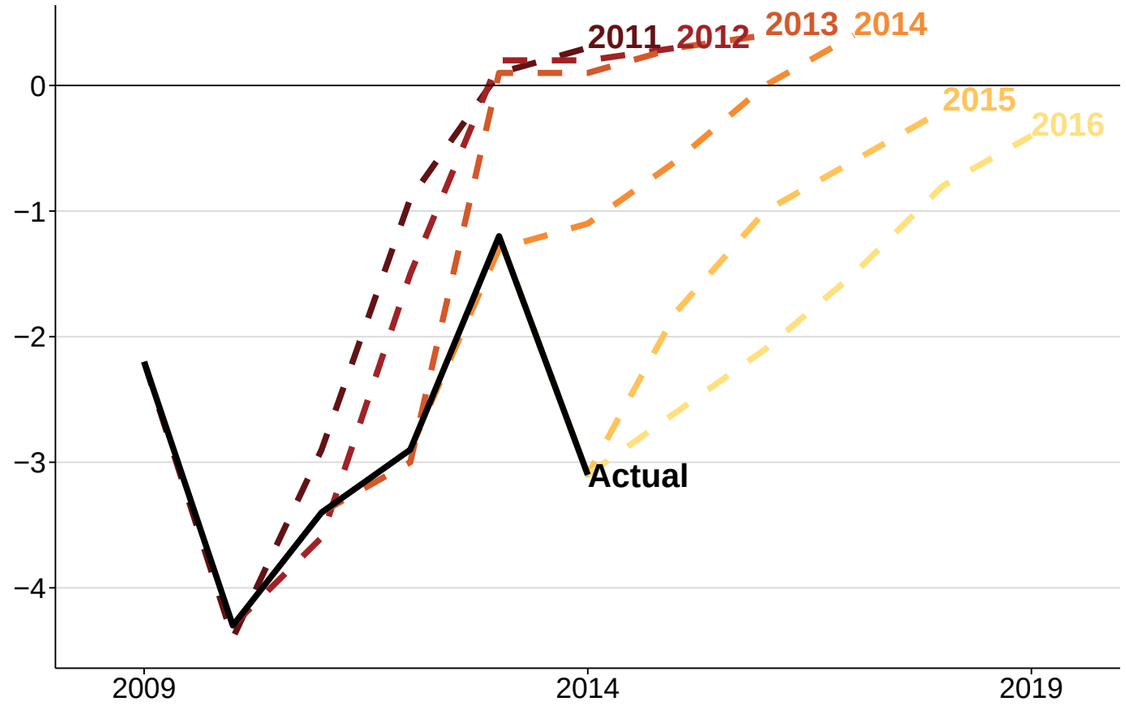
```
## Warning:  Removed 56 rows containing missing values (geom_text).
```



1.3 Forecast charts

```
read.table(text="Year of forecast 2011 2012 2013 2014 2015 2016 Actual
2009 -2.2 -2.2
2010 -4.4 -4.3 -4.3
2011 -2.9 -3.6 -3.4 -3.4
2012 -0.9 -1.5 -3 -2.9 -2.9
2013 0.1 0.2 0.1 -1.3 -1.2 -1.2
2014 0.3 0.2 0.1 -1.1 -3.1 -3.1 -3.1
2015 0.3 0.3 -0.6 -1.8 -2.6
2016 0.4 0 -1 -2.1
2017 0.4 -0.6 -1.5
2018 -0.2 -0.8
2019 -0.4
", sep = "\t", header = TRUE, check.names = FALSE) %>%
  # the `Year of forecast` refers to the row, not the column:
  tidyr::gather(Year_of_forecast, value, -`Year of forecast`) %>%
  rename(Year_date = `Year of forecast`) %>%
  mutate(Year_of_forecast = factor(Year_of_forecast, levels = c(2011:2016, "Actual"))) %>%
  group_by(Year_of_forecast) %>%
  mutate(Year_of_forecast.label = ifelse(Year_date == max(Year_date[!is.na(value)]),
                                         as.character(Year_of_forecast),
                                         NA_character_)) %>%

  {
    grplot(., aes(x = Year_date, y = value, color = Year_of_forecast)) +
      # We want to plot actual over the top of the others:
      geom_line(data = filter(., Year_of_forecast != "Actual"),
                linetype = "dashed") +
      geom_line(data = filter(., Year_of_forecast == "Actual"), color = "black") +
      geom_text(data = filter(., Year_of_forecast != "Actual"),
                aes(label = Year_of_forecast.label),
                hjust = 0,
                vjust = 0,
                size = 23/(14/5),
                fontface = "bold") +
      geom_text(data = filter(., Year_of_forecast == "Actual"),
                aes(label = Year_of_forecast.label),
                hjust = 0,
                size = 23/(14/5),
                color = "black",
                fontface = "bold") +
      scale_x_continuous(limits = c(2008, 2020),
                         breaks = c(2009, 2014, 2019),
                         expand = c(0,0)) +
      theme(axis.title.x = element_blank()) +
      geom_hline(yintercept = 0)
  }
```

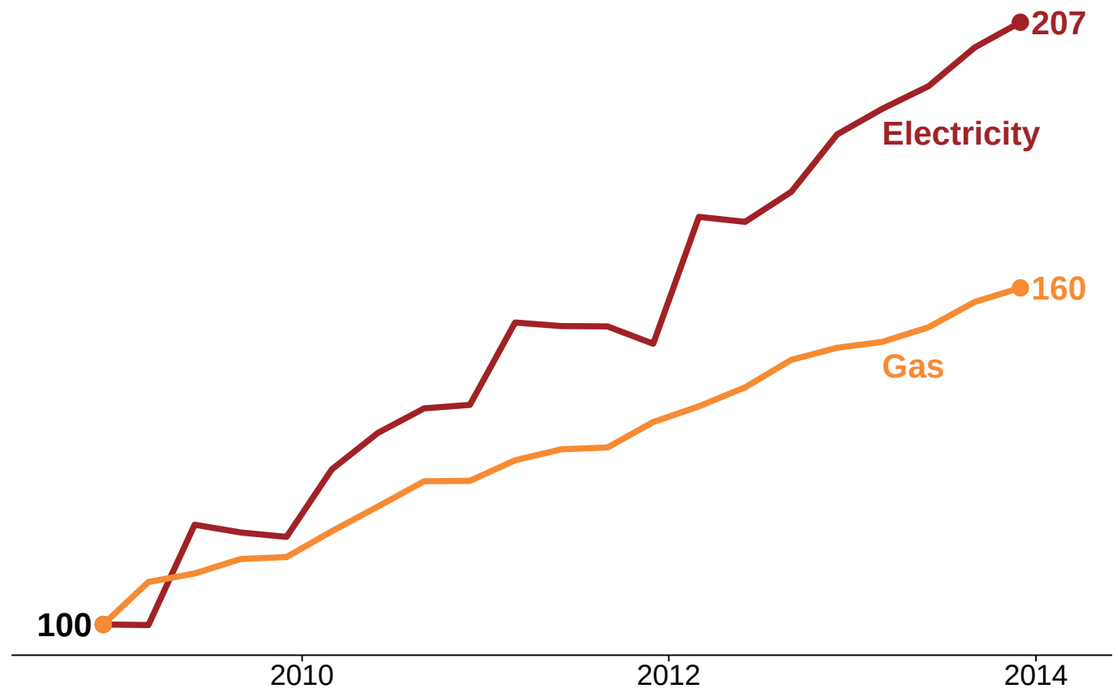
```
## Warning:  Removed 30 rows containing missing values (geom_path).
## Warning:  Removed 5 rows containing missing values (geom_path).
## Warning:  Removed 60 rows containing missing values (geom_text).
## Warning:  Removed 10 rows containing missing values (geom_text).
```



1.4 Index charts

```
# generate data
set.seed(1)
data_frame(Date = seq(as.Date("2008-12-01"), as.Date("2013-12-01"),
                      by = "3 months")) %>%
  mutate(Electricity = rnorm(1, 50) + 3 * 1:n() + rnorm(n(), 5, 3),
         Gas = rnorm(1, 50) + 1.5 * 1:n() + rnorm(n(), 5), 3) %>%
  tidyr::gather(Source, value, Electricity:Gas) %>%
  group_by(Source) %>%
  mutate(index = 100*value/first(value),
         index.label = ifelse(value == last(value),
                              round(index),
                              NA_real_)) %>%
  mutate(Source.label = ifelse(year(Date) == 2013 & month(Date) == 3,
                               as.character(Source),
                               NA_character_)) %>%
{
  grplot(., aes(x = Date, y = index, color = Source)) +
    geom_line() +
    scale_color_manual(values = gpal(2, dark = TRUE)) +
    theme(axis.title.x = element_blank()) +
    geom_text(aes(label = index.label),
              hjust = -0.2,
              fontface = "bold",
              size = 23/(14/5)) +
    geom_text(aes(label = Source.label),
              vjust = 1.5,
              hjust = 0,
              fontface = "bold",
              size = 23/(14/5)) +
    geom_point(data = filter(ungroup(.), Date == max(Date) | Date == min(Date)),
               size = 5) +
    annotate("text", x = as.Date("2008-12-01"), y = 100, hjust = 1.2, label = "100",
             fontface = "bold", size = 23/(14/5)) +
    theme(axis.text.y = element_blank(), axis.text.y = element_blank(),
          panel.grid.major = element_blank(),
          axis.ticks.y = element_blank(), axis.line.y = element_blank()) +
    scale_x_date(expand = c(0.1, 0.1))
}
```

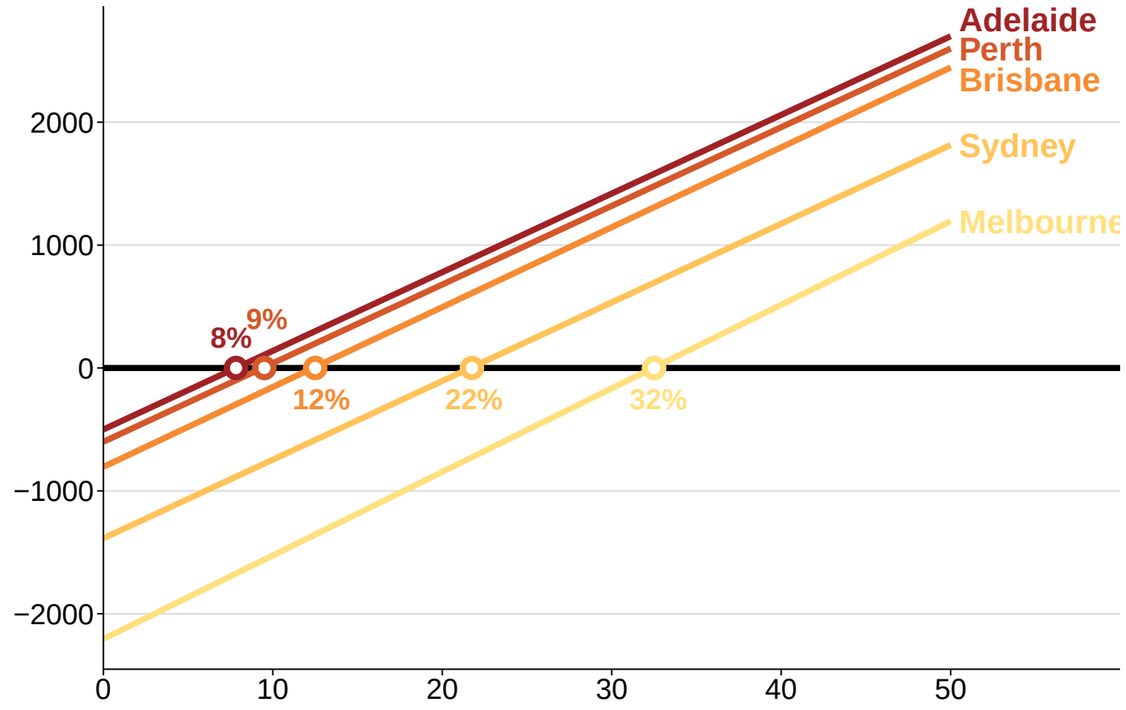
```
## Warning:  Removed 40 rows containing missing values (geom_text).
## Warning:  Removed 40 rows containing missing values (geom_text).
```



1.5 Annotating lines

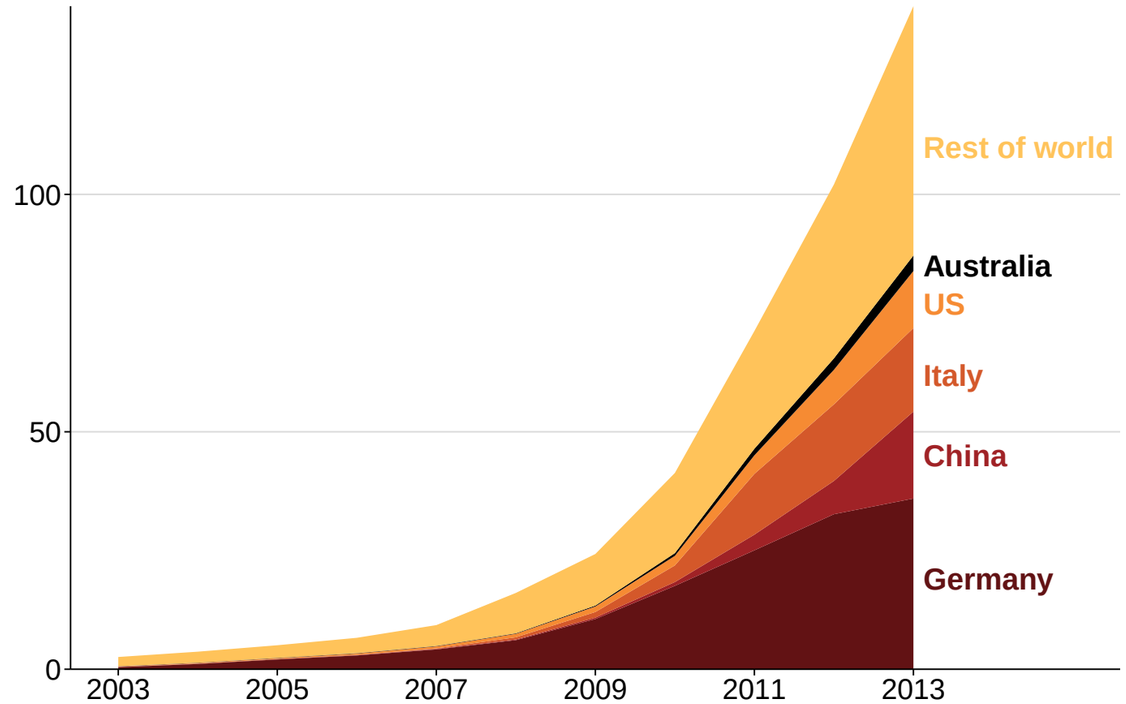
```
read.table(text=[1352 chars quoted with '"'], header=TRUE) %>%
  tidyr::gather(City, value, -X.Values) %>%
  filter(City %in% c("Adelaide", "Perth", "Brisbane", "Sydney", "Melbourne")) %>%
  mutate(City = factor(City, levels = c("Adelaide", "Perth", "Brisbane", "Sydney", "Melbourne"))) %>%
  group_by(City) %>%
  mutate(City.label = ifelse(X.Values == max(X.Values),
                             as.character(City),
                             NA_character_)) %>%
  ungroup %>%
  {
    grplot(., aes(x = X.Values, y = value, color = City)) +
      # coloured lines (i.e. data) should go *over* y = 0, so
      # plot y = 0 first
      geom_hline(yintercept = 0, size = 2, color = "black") +
      geom_line() +
      geom_text(aes(label = City.label,
                    # avoid collisions
                    vjust = ifelse(City == "Adelaide",
                                   -0.2, ifelse(City == "Brisbane",
                                                1,
                                                0.5))),
                nudge_x = 0.5,
                hjust = 0,
                fontface = "bold",
                size = 23/(14/5)) +
      scale_x_continuous(expand = c(0,0), limits = c(0, max(.$X.Values) * 1.2),
                         breaks = 10*(0:5)) +
      theme(axis.title.x = element_blank()) +
      annotate("point",
               # ordinarily this would be present in the data
               x = c(7.8275, 9.5, 12.5, 21.75, 32.5),
               y = 0,
               shape = 21, # hollow point
               stroke = 3,
               fill = "white",
               color = gpal(5),
               size = 4) +
      annotate("text",
               x = c(7.55, 9.65, 12.875, 21.875, 32.75),
               vjust = 0.5,
               fontface = "bold",
               size = 20/(14/5),
               color = gpal(5),
               y = c(250, 400, -250, -250, -250),
               label = percent(c(8, 9, 12, 22, 32)/100))
  }
```

```
## Warning:  Removed 80 rows containing missing values (geom_text).
```



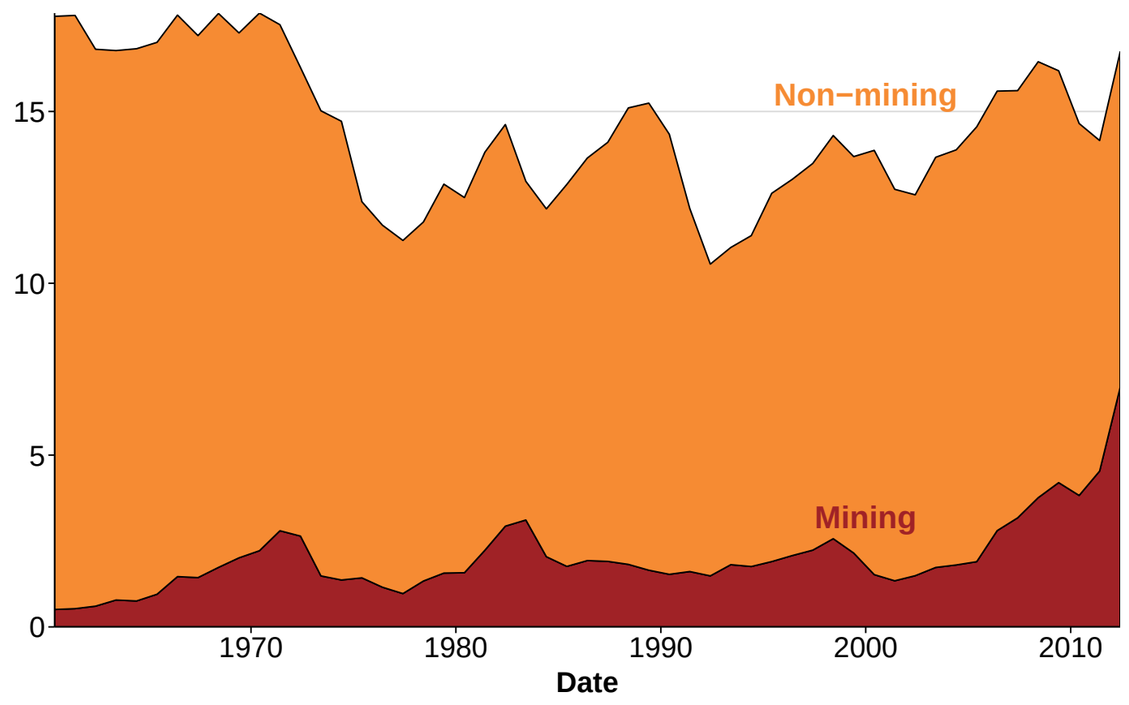
2 Area charts

```
read.table(text="Year Germany China Italy US Australia Rest of World
2003 0.435 0.055 0.026 0.073 0.0456 1.940809
2004 1.105 0.064 0.0307 0.131 0.0523 2.315009
2005 2.056 0.068 0.0375 0.172 0.0606 2.654309
2006 2.899 0.0799 0.05 0.275 0.0703 3.244323
2007 4.17 0.0999 0.1202 0.427 0.0825 4.39135811
2008 6.12 0.1399 0.4583 0.738 0.1045 8.50235811
2009 10.566 0.2999 1.1813 1.172 0.1876 10.85800811
2010 17.554 0.7999 3.5023 2.022 0.5709 16.88070811
2011 25.039 3.2999 12.8029 3.91 1.3768 24.7890373
2012 32.643 7 16.139 7.271 2.407 36.61577044
2013 35.948 18.3 17.6 12.022 3.255 52.51189856
", header = TRUE, sep = "\t") %>%
  tidyr::gather(Country, value, -Year, factor_key = TRUE) %>%
  grplot(aes(x = Year, y = value, fill = Country)) +
  geom_area() +
  scale_fill_manual(values = c(gpal(6)[1:4], "black", gpal(6)[5])) +
  scale_x_continuous(breaks = seq(2003, 2013, by = 2),
                     limits = c(2003, 2015)) +
  theme(axis.title = element_blank()) +
  annotate("text",
           x = 2013.125,
           y = c(19, 45, 62, 77, 85, 110),
           hjust = 0,
           fontface = "bold",
           size = 21/(14/5),
           color = c(gpal(6)[1:4], "black", gpal(6)[5]),
           label = c("Germany", "China", "Italy", "US", "Australia", "Rest of world")) +
scale_y_continuous(expand = c(0,0))
```



2.1 Another area chart

```
read_csv("Supp-Data/Mining-area-chart.csv") %>%
  tidyr::gather(Sector, value, -Date) %>%
  mutate(Date = as.Date(paste0("01-", Date), format = "%d-%b-%Y")) %>%
  grplot(aes(x = Date, y = value, fill = Sector)) +
  # set color = 'black' to put a border around the area
  geom_area(color = "black") +
  scale_fill_manual(values = gpal(2, TRUE)) +
  annotate("text",
           x = as.Date("2000-01-01"),
           y = c(3.2, 15.5),
           label = c("Mining", "Non-mining"),
           size = 22/(14/5),
           color = gpal(2, dark = TRUE),
           fontface = "bold") +
  scale_x_date(expand = c(0,0)) +
  scale_y_continuous(expand = c(0,0))
```

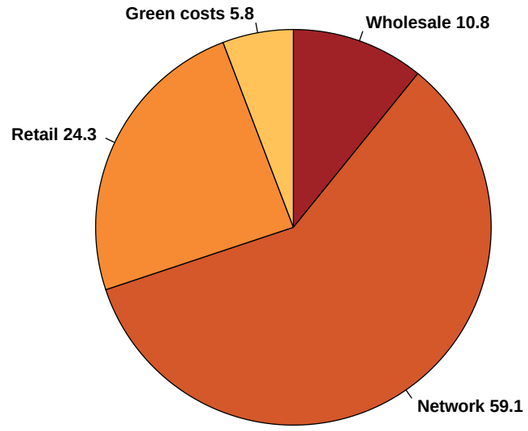


2.2 Overlapping area charts

```
read_excel("Supp-Data/Solar-usage.xlsx") %>%
  data.table::setnames(1:4, c("Time", "Consumption", "Solar2", "Solar4")) %>%
  gather(Type, value, -Time) %>%
  grplot(aes(x = Time, y = value, fill = Type)) +
  geom_area(position = "dodge", color = "black") +
  scale_x_datetime(date_labels = )

## Warning:  Width not defined.  Set with `position_dodge(width = ?)`
```

2.3 Pie charts

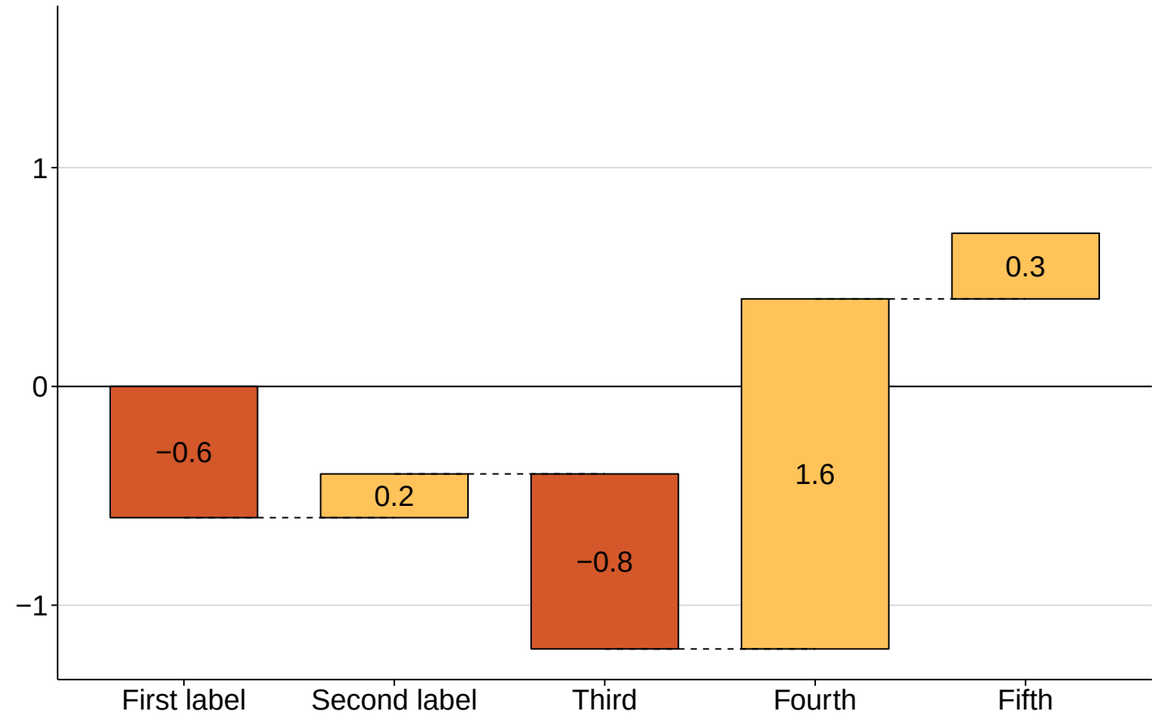
```
read.table(text="Sector NSW
Wholesale 10.83210129
Network 59.05925758
Retail 24.31851783
Green costs 5.790123293
", header = TRUE, sep = "\t") %$%
  # base graphics!  Leaves a fair bit of margin. This
  # is ultimately unavoidable for pie charts, but can
  # be improved if required.
  pie(NSW, labels = paste(.$Sector, round(.$NSW, 1)),
      col = gpal(nrow(.)),
      font = 2,  #bf
      clockwise = TRUE)
```



2.4 Waterfall charts

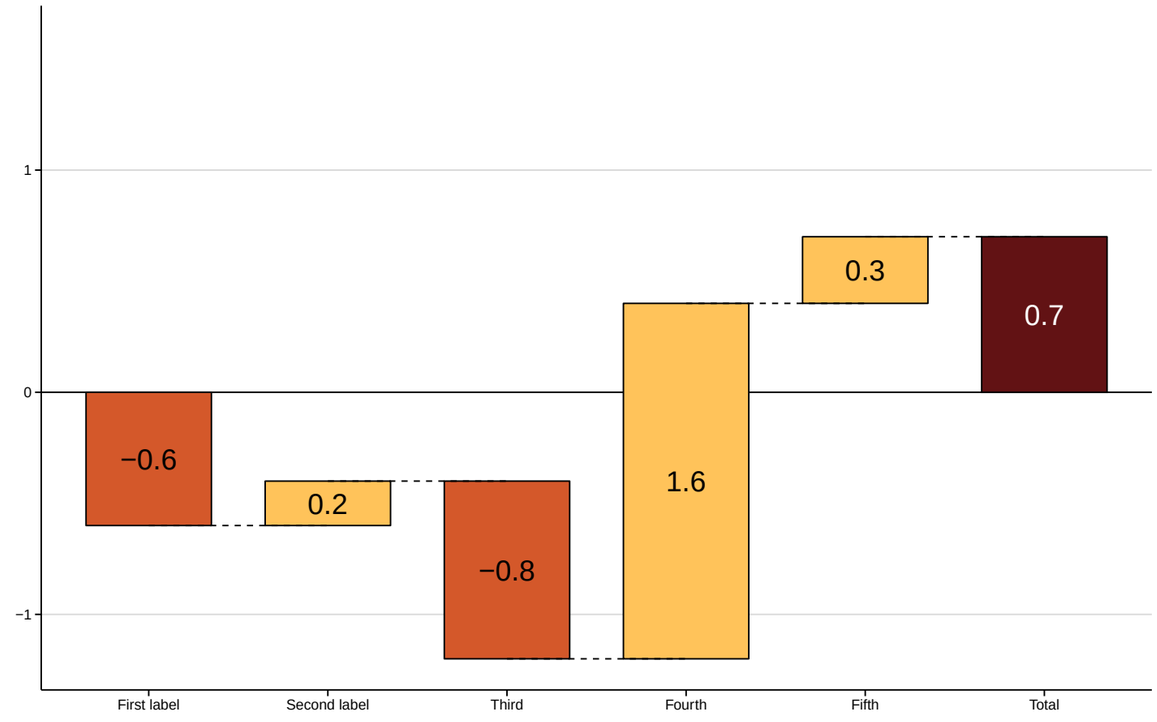
2.4.1 Plain

```
set.seed(1)
grattan_waterfall(values = round(rnorm(5), 1),
                  labels = c("First label", "Second label", "Third", "Fourth", "Fifth"),
                  print_plot = FALSE) +
  theme_hugh(base_size = 20) +
  theme(axis.title.x = element_blank())
```



2.4.2 With totals

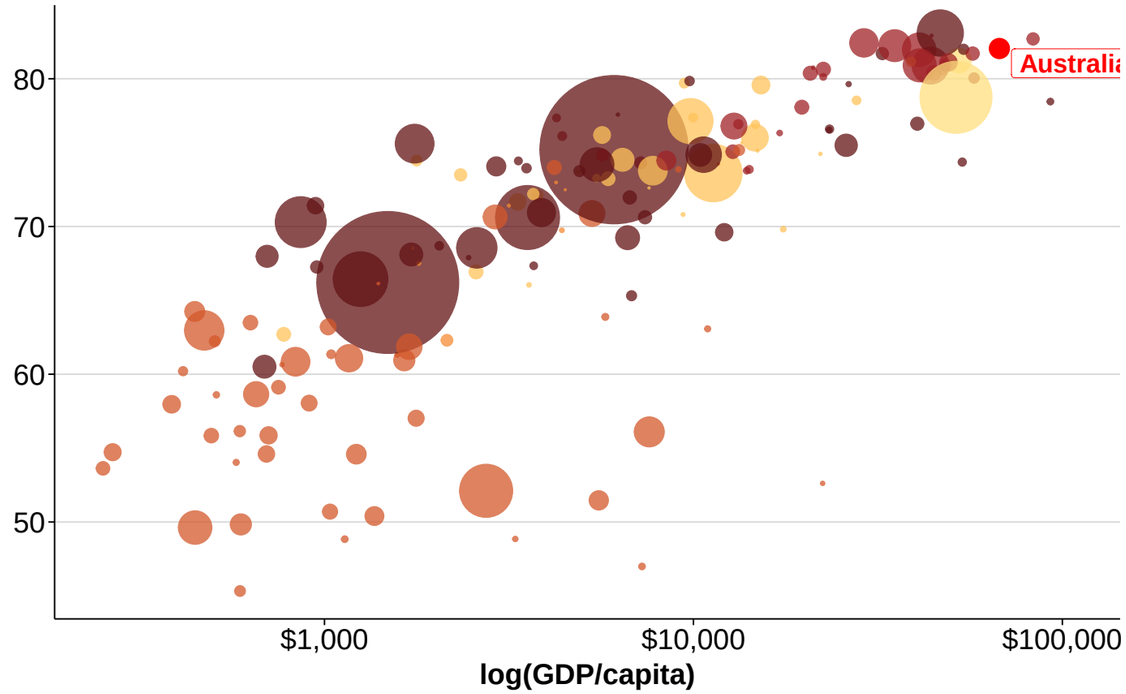
```
set.seed(1)
grattan_waterfall(values = round(rnorm(5), 1),
                  labels = c("First label", "Second label", "Third", "Fourth", "Fifth"),
                  #
                  calc_total = TRUE)
```



2.5 Bubble chart

```
temp.data <-
  fread("./Supp-Data/bubble-chart.csv")

## Warning in fread("./Supp-Data/bubble-chart.csv"):  Unable to find 5 lines with expected number of
columns (+ middle)
## Warning in fread("./Supp-Data/bubble-chart.csv"):  Unable to find 5 lines with expected number of
columns (+  last)

temp.data <-
  read_excel("./Supp-Data/bubble-chart.xlsx")

for (j in seq_along(names(temp.data)))
  set(temp.data, j = j, value = ifelse(temp.data[[j]] == -99, NA_real_, temp.data[[j]]))


temp.data %>%
  select(GDP_per_capita = `GDP/capita ($US)`,
         Life_expectancy = `Life expectancy`,
         Continent,
         Population, Country) %>%
{
  grplot(., aes(x = GDP_per_capita,
                y = Life_expectancy,
                size = Population,
                color = factor(Continent))) +
    geom_point(alpha = 0.75) +
    geom_point(data = filter(., Country == "Australia"), color = "red") +
    geom_label(data = filter(., Country == "Australia"),
               mapping = aes(label = Country),
               colour = "red",
               hjust = -0.10,
               fontface = "bold",
               vjust = 0.5,
               fill = "white",
               nudge_y = -1) +
    scale_x_log10("log(GDP/capita)", label = grattan_dollar) +
    scale_size_area(max_size = 48)
}
```

```
## Warning:  Removed 67 rows containing missing values (geom_point).
```



2.6 Facetted area chart

```
read_excel("./Supp-Data/Income-1976--2011-by-age_group.xlsx", sheet = 2) %>%
  select(Age_range, Incomes, Year) %>%
  filter(complete.cases(.)) %>%
  group_by(Age_range) %>%
  arrange(Year) %>%
  mutate(goes_up = last(Incomes) > first(Incomes)) %>%
  mutate(text.label = ifelse(Age_range == "25-34" & (Year == 1976 | Year == 2011),
                             Year,
                             NA_character_)) %>%
  grplot(aes(x = Year, y = Incomes)) +
  geom_area(fill = col.1, alpha = 0.8) +
  geom_line(aes(color = goes_up)) +
  scale_color_manual(values = gpal(2, TRUE)) +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.title.x = element_blank()) +
  geom_text(aes(label = text.label,
                hjust = ifelse(Year == 1976, 0.1, 0.9)),
            size = 20/(14/5),
            nudge_y = 1000,
            vjust = 0) +
  facet_grid(~Age_range, switch = "x") +
  theme(strip.background = element_blank(),
        strip.text = element_text(size = 23)) +
  scale_y_continuous(limits = c(0,50e3), expand = c(0,0), label = grattan_dollar)

# Absolute positioning
grid.text("Income decline", x = 0.15, y = 0.9, gp = gpar(col = gpal(2, TRUE)[1], font = 2, fontsize = 22), hjust = 0)
grid.text("Income increase", x = 0.65, y = 0.9, gp = gpar(col = gpal(2, TRUE)[2], font = 2, fontsize = 22), hjust = 0)
```