

Taking control of graphics using R

Yvan Richard
yvan@dragonfly.co.nz


Wellington R-Users Group
17 March 2014



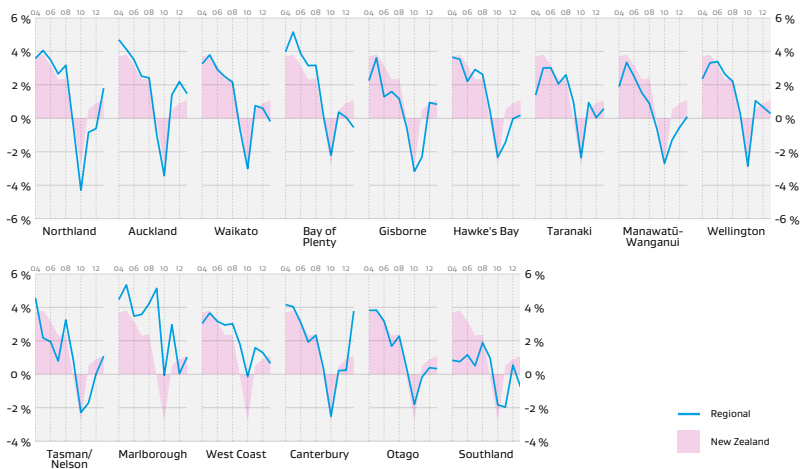
Outline

- 1 Examples of advanced plots in pure R
- 2 Improving from R base graphics
- 3 Data
- 4 Plotting
- 5 Subplots
- 6 Styling

Some examples of plots I created for the New Zealand Ministry of Business, Innovation & Employment

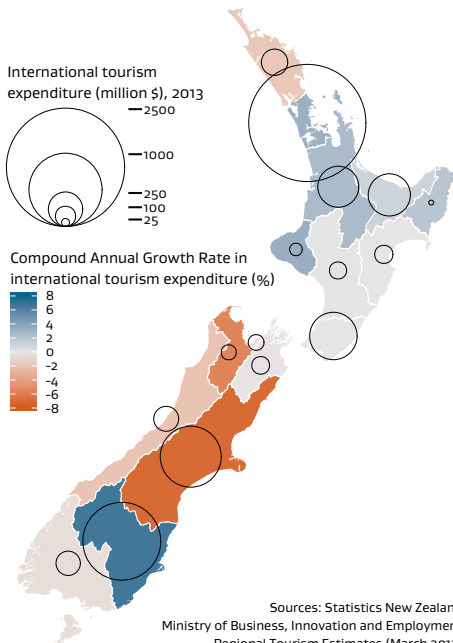
- All made using programming only (except logos)
- Only using 
- The scripts will be made public after report completion

Annual employment growth in the regions versus New Zealand - 2003-2013



Source: Employment Estimates, MBIE

Compound Annual Growth Rate in international tourism expenditure, 2009-2013



Current travel time and GDP growth

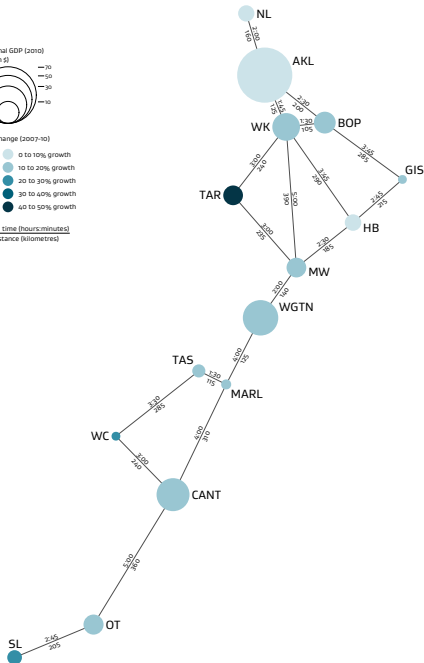
Regional GDP (2010)
(billion \$)



GDP change (2007-10)



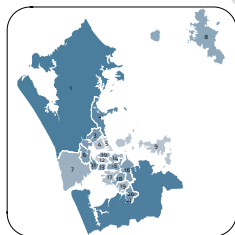
Travel time (hours:minutes)
Distance (kilometres)



Note: Separation of regional centres is proportional to travel time

Source: New Zealand Transport Agency,
Automobile Association, Statistics New Zealand

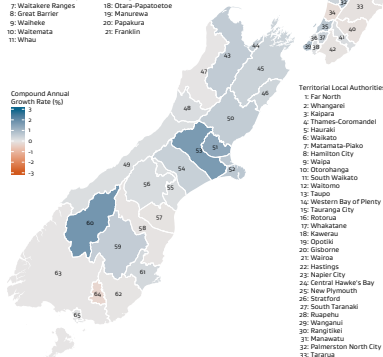
Average annual growth (%) in population by TLA - 2013-2031



Auckland Local Board Areas

- | | |
|-----------------------|-------------------------|
| 1: Rodney | 12: Albert-Eden |
| 2: Hibiscus and Bays | 13: Puketāpapa |
| 3: Upper Harbour | 14: Orakei |
| 4: Kaipātiki | 15: Maungakiekie-Tamaki |
| 5: Devonport-Takapuna | 16: Howick |
| 6: Henderson-Massey | 17: Māngero-Otahuhu |
| 7: Waiākare Ranges | 18: Ōtara-Papatoetoe |
| 8: Great Barrier | 19: Manurewa |
| 9: Waiheke | 20: Papakura |
| 10: Waiwemata | 21: Franklin |
| 11: Whau | |

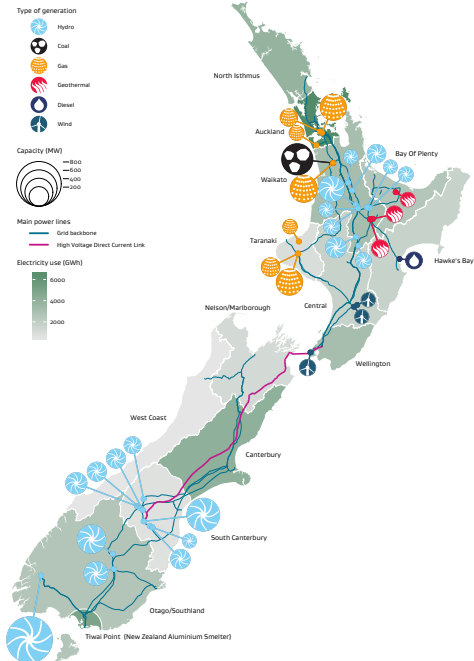
Compound Annual Growth Rate (%)



Territorial Local Authorities

- | | |
|---------------------------|-----------------------|
| 1: Far North | 34: Horowhenua |
| 2: Whangarei | 35: Kapiti Coast |
| 3: Kaipara | 36: Porirua City |
| 4: Thames-Coromandel | 37: Upper Hutt City |
| 5: Hauraki | 38: Lower Hutt City |
| 6: Waikato | 39: Wellington City |
| 7: Matamata-Piako | 40: Masterton |
| 8: Hamilton City | 41: Carterton |
| 9: Waipa | 42: South Wairarapa |
| 10: Otorohanga | 43: Tasman |
| 11: South Waikato | 44: Nelson City |
| 12: Waikato | 45: Marlborough |
| 13: Taupo | 46: Kaikoura |
| 14: Western Bay of Plenty | 47: Buller |
| 15: Tauranga City | 48: Grey |
| 16: Rotorua | 49: Westland |
| 17: Whakatane | 50: Hurunui |
| 18: Kawerau | 51: Waimakariri |
| 19: Opotiki | 52: Christchurch City |
| 20: Gisborne | 53: Selwyn |
| 21: Waimea | 54: Ashburton |
| 22: Hastings | 55: Timaru |
| 23: Napier City | 56: Mackenzie |
| 24: Central Hawke's Bay | 57: Waimate |
| 25: New Plymouth | 58: Waikati |
| 26: Stratford | 59: Central Otago |
| 27: South Taranaki | 60: Queenstown-Lakes |
| 28: Ruapehu | 61: Dunedin City |
| 29: Wangarua | 62: Clutha |
| 30: Rangitikei | 63: Southland |
| 31: Manawatu | 64: Gore |
| 32: Palmerston North City | 65: Invercargill City |
| 33: Taranaki | 66: Auckland |

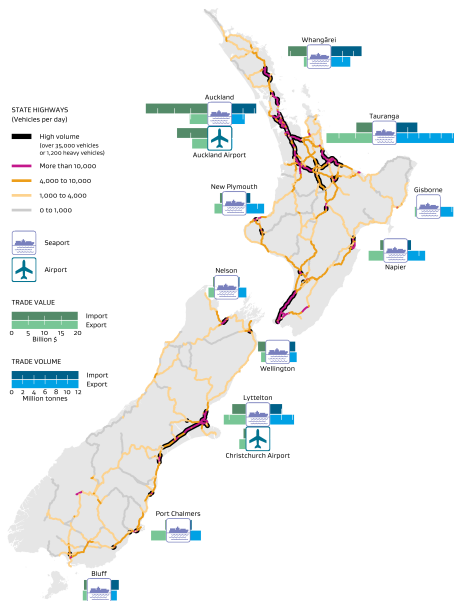
Electricity infrastructure and use by region - 2013



Note: Only the largest power plants are represented, accounting for 90% of the national production

Source: Electricity Authority, Transpower New Zealand

Major New Zealand transport network connections 2013



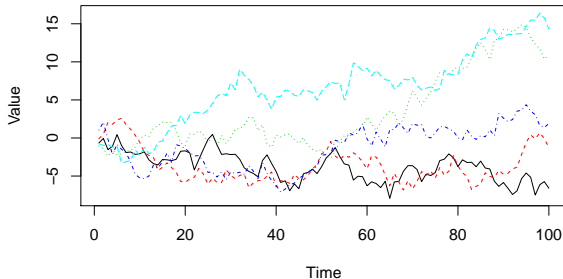
Programming vs. hand drawing

Programming may seem tedious but

- Plots are easy to update with new data
- Code is easy to re-use and adapt (use **GNU make** for work flow management)
- Easily scalable (no difference between 5 regions or 10,000)
- Changes are easy to track (use **GIT** for version control)
- Transparent
- Guarantee that the plot reflects the data
 - No hand tweaking of e.g. bubble sizes or region colours
 - What you see is directly derived from the original data

Simple plots in R are quick and easy, but lack finesse...

```
par(mar=c(4,4,.5,.5))  
matplot(apply(matrix(rnorm(500)), nrow=5), 1, cumsum),  
        xlab='Time', ylab='Value', type='l')
```

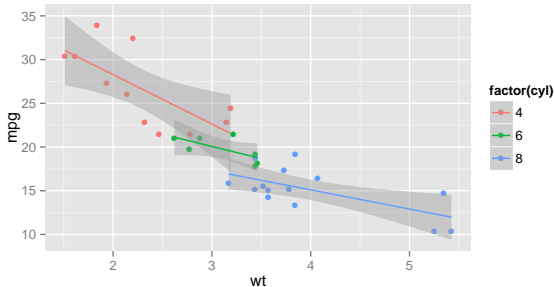


Improved R graphics

Improving from R base graphics

ggplot2 provides a consistent visual style (grammar of graphics), but it constrains you.

```
library(ggplot2)
ggplot(mtcars, aes(x=wt, y=mpg, col=factor(cyl))) + geom_point() +
  geom_smooth(aes(group=cyl), method='lm') +
  theme(plot.margin=unit(c(0,0,0,0), 'line'))
```



ggplot2 can still be used for making customised graphics, especially by using the power of grid graphics

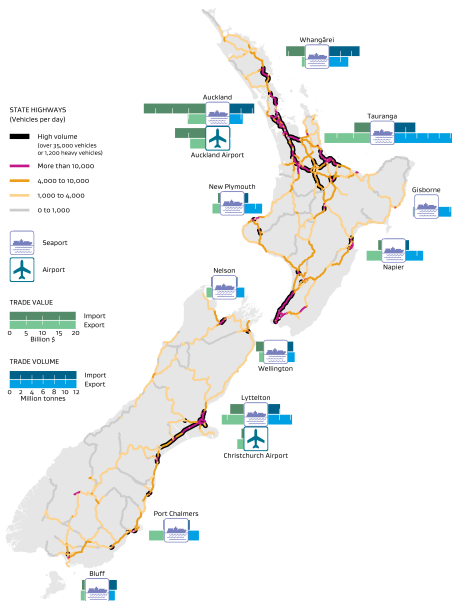
Customisation is needed when:

- A specific style or design is required (client, journal, organisation, ...)
- Fitting a lot of information in a single figure
- Plots are infographics

Many libraries are available. See
<http://cran.r-project.org/web/views/Graphics.html>

Focus here on ggplot2 and grid, and a single plot

Major New Zealand transport network connections 2013



Several sources:

- GIS shapefiles (NZ coastline, road networks)
- Spreadsheet with port locations and position tweaks (manual)
- Spreadsheets with import/export values and volumes

```
library(rgdal)
nz <- readOGR("gis_folder", "nz-coastline",
              p4s="+init=epsg:4326") ## lat/long projection
## Project to NZTM (in metres)
nz <- spTransform(nz, CRS("+init=epsg:2193"))

## Fortify - Converts spatial object to a data frame of
## x and y coordinates usable by ggplot
library(ggplot2)
nz <- fortify(nz)
save(nz, file='nz-coastline-nztm.rdata')
```


Spreadsheets

Data

	1	2	3	4	5	6
1	location	x	y	offset_x	offset_y	name_dir
2	Auckland Airport	174.763332	-36.84846	-150000	-45000	B
3	Auckland	174.763332	-36.84846	-150000	15000	T
4	Christchurch Airport	172.636225	-43.532054	130000	-40000	B
5	Lyttelton	172.636225	-43.532054	130000	20000	T
6	Dunedin Airport	170.502798	-45.87876	0	0	T
7	Port Chalmers	170.502798	-45.87876	100000	0	T
8	Gisborne	178.017649	-38.662334	80000	0	T
9	Greymouth	171.210762	-42.450392	0	0	T
10	Hamilton Airport	175.279253	-37.787001	0	0	T
11	Invercargill Airport	168.353773	-46.413187	0	0	T
12	Bluff	168.353773	-46.413187	70000	-80000	T
13	Napier	176.912018	-39.492844	100000	-20000	B
14	Nelson	173.283965	-41.270632	0	80000	T
15	New Plymouth	174.075228	-39.055625	-50000	40000	T

```
ports_loc <- read.csv('ports-locations.csv', as.is=T)

library(sp)
coordinates(ports_loc) <- ~ x + y
proj4string(ports_loc) <- "+init=epsg:4326"
ports_loc <- as.data.frame(spTransform(ports_loc, CRS("+init=epsg:2193")))
```

Normalise data to have only one row per “point”

Not:

	1	2	3	4	5
1	Region	Year_2008	Year_2009	Year_2010	Year_2011
2	Auckland	23548	21543	23512	26128
3	Wellington	10256	11254	13985	12984

But:

	1	2	3
1	Region	year	value
2	Auckland	Year_2008	23548
3	Wellington	Year_2008	10256
4	Auckland	Year_2009	21543
5	Wellington	Year_2009	11254
6	Auckland	Year_2010	23512
7	Wellington	Year_2010	13985
8	Auckland	Year_2011	26128
9	Wellington	Year_2011	12984

```
library(reshape2)
melt(data, id.vars='Region', variable.name='year')
```

Plot NZ coastline, and road networks:

```
p <- qplot(xlims, ylims, geom = "blank") +  
  geom_polygon(aes(x=long, y=lat, group=id), fill=colour_nz, colour=NA,  
               data=nz) +  
  geom_path(aes(x=long, y=lat, group=group), data=hightraff,  
            size=2.5*size_large_roads, colour='black') +  
  geom_path(aes(x=long, y=lat, group=group, colour=traff), data=traff,  
            size=size_large_roads) +  
  scale_colour_manual(values = routes_cols, guide='none') +  
  coord_fixed(ratio=1, xlim=xlims, ylim=ylims) +  
  theme_blank()
```

`xlims`, `ylims`, `colour_nz`, `size_large_roads` are variables defined at the beginning of script

`routes_cols` is a named vector relating discrete values of route traffic to their colour

Subplot creation

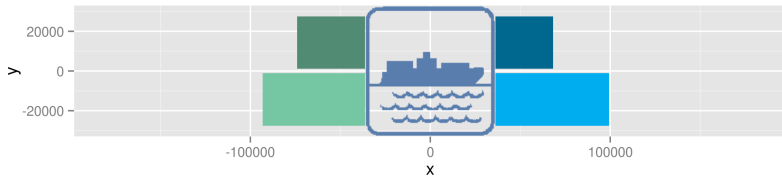
Subplots



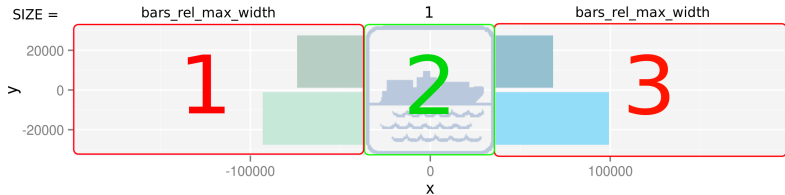
Subplot creation

Subplots

Without theme



Three viewports



```
boat_grob <- rasterGrob(readPNG('boat-logo.png'))

pushViewport(
  viewport(layout = grid.layout(1, 3,
    widths = unit(c(bars_rel_max_width, 1, bars_rel_max_width),
      rep('grobwidth', 3), list(boat_grob, boat_grob, boat_grob)),
    heights = grobHeight(boat_grob))))
```

One of the two bar plots:

```
bars_vol <- ggplot(vol, aes(x=import_export, y=value, fill=import_export)) +  
  geom_bar(stat='identity', width=1-gap_betweenBars) +  
  scale_fill_manual(guide='none', values=c(import=colour_bar_imp_vol,  
                                            export=colour_bar_exp_vol)) +  
  coord_flip(ylim=c(0, max.vol)) + # make bars horizontal and set limits  
  scale_y_continuous(expand=c(0,0)) + # remove gaps  
  theme_nothing()
```

Add it to viewport 3:

```
pushViewport(viewport(layout.pos.row = 1, layout.pos.col = 3))  
grid.draw(ggplotGrob(bars_vol))  
upViewport()
```

Adding subplot to main plot

Subplots



```
subplot <- grid.grabExpr({  
  pushViewport(...)  
  grid.draw(...)  
  ... })  
  
subplot_width <- (2*bars_rel_max_width + 1)*subplot_size  
  
p1 <- p + annotation_custom(subplot,  
  xmin = loc_x - subplot_width/2, xmax = loc_x + subplot_width/2,  
  ymin = loc_y - subplot_size/2, ymax = loc_y + subplot_size/2)
```


Adding grid object in user coordinates

Subplots

Example: add a reference scale for the bar plots (a rectangle the width of the longest bar)

```
g <- ggplot(...)
print(g)

downViewport('panel.3-4-3-4')
gb <- ggplot_build(g)
pushViewport(dataViewport(xscale=gb$panel$ranges[[1]]$x.range,
                          yscale=gb$panel$ranges[[1]]$y.range,
                          clip='off'))

grid.rect(x = unit(x_coord, 'native'),
          y = unit(y_coord, 'native'),
          width = unit(subplot_size * bars_rel_max_width, 'native'),
          height = unit(0.5, 'cm'))
```

Adding grid object in user coordinates

Subplots

Grid offers many primitive shapes for adding to plots

- `grid.rect()`, `grid.lines()`, `grid.text()`, `grid.arrows()`, ...

Different units can be conveniently combined (i.e., for adding spacing):

```
grid.text(x = unit(coord_x, 'native') + unit(3, 'mm'), ...)
```

Use theme

```
ggplot(...) +  
  theme(axis.ticks.length = unit(1.5, "mm"),  
        panel.background = element_blank())
```

Or replace geoms with your own function to use consistent formats:

```
title_text <- function(...)  
  geom_text(..., size=10, family='Helvetica', lineheight=0.8)  
  
ggplot(...) + title_text('This is a title')
```

Use gpar

```
grid.rect(gp = gpar(col='red', fill=NA, lwd=2))
```

Also make use of functions for consistency

```
gpar_title <- function(...)  
  gpar(fontsize=10, fontfamily='Helvetica', lineheight=0.8, ...)  
  
grid.text('This is a title', gp=gpar_title())
```

Thank you

This presentation and a self-contained project for making the main plot are publicly available on GitHub:

<https://github.com/dragonfly-science/r-users-group-presentation.git>

Thank you to the whole R and open-source community for making this possible.

Main software used: R, LaTeX, Beamer, GNU make, GIT, Emacs