Simple Plots

Displaying a Simple Batch of Numbers

- For a very small set of numbers ($n \leq 3$), it is best to simply list the values.

$$0.32 \quad 0.95 \quad 0.73$$

Displaying a Simple Batch of Numbers

- For a very small set of numbers ($n \leq 3$), it is best to simply list the values.

$$0.32 \quad 0.95 \quad 0.73$$

- For slightly larger sets of values ($n \leq 20$), listing the values in order can provide some insight.

-1.4	-1.3	-1.1	-0.8	-0.7
-0.4	-0.4	-0.3	-0.2	-0.2
-0.2	-0.1	-0.1	0.0	0.1
0.1	0.7	0.8	0.9	1.8

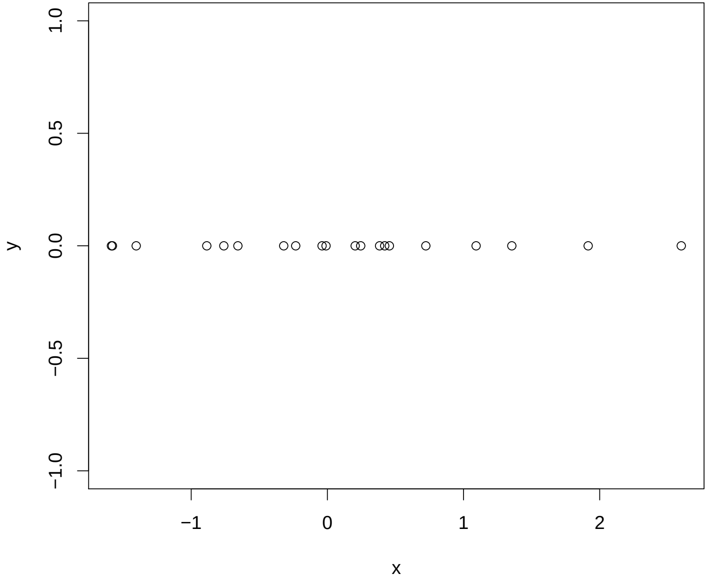
Plotting a Simple Batch of Numbers

- Any graphical encoding method (area, angle, length, postition, ...) can be used to display the values.

- The *best* encoding method is position on common scale and this should be the basis of any plotting method.

- The simplest method is produce a one dimensional scatterplot or *stripchart*.

```
> x = rnorm(20)
> y = rep(0, length(x))
> plot(x, y)
```


1.0
0.5
0.0
−0.5
−1.0
y
−1
0
1
2
x

Improved Stripcharts

There are clear problems with the previous plot.

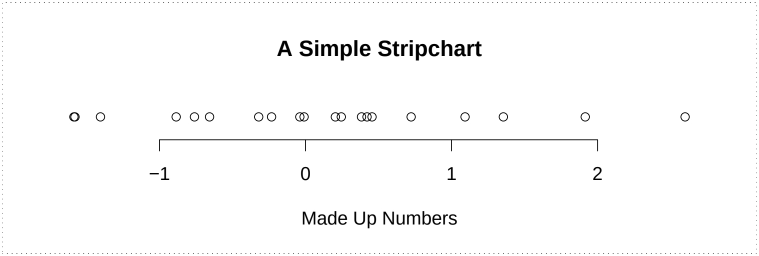
- The use of vertical space is wasteful.
- Having a y axis in the plot makes no sense.
- The margins used for a typical scatterplot are not appropriate.

Improved Stripchart Code

Here is how we can create a suitable plot. The code below is appropriate for a a device size of 9×3 inches and a font pointsize of 16. Scaling all these values by the same amount should produce a similar plot.

```
> par(mar = c(5.1, 2.1, 4.1, 2.1))
> plot.new()
> plot.window(xlim = range(x),
              ylim = c(-1, 1))
> points(x, y)
> axis(1)
> title(main = "A Simple Stripchart")
> title(xlab = "Made Up Numbers")
```

A Simple Stripchart

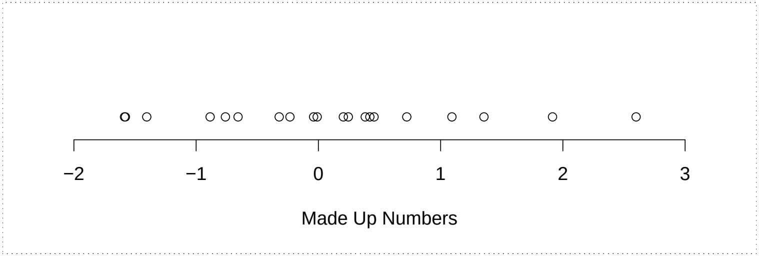


Expanded Axes for Stripcharts

Here a call to `pretty` has been added. This ensures that the outermost tickmarks on the x axis lie outside the extreme data values.

```
> par(mar = c(5.1, 2.1, 4.1, 2.1))
> plot.new()
> plot.window(xlim = range(pretty(x)),
              ylim = c(-1, 1))
> points(x, y)
> axis(1)
> title(xlab = "Made Up Numbers")
```

A Stripchart with Exanded Axis



Dealing With Overplotting

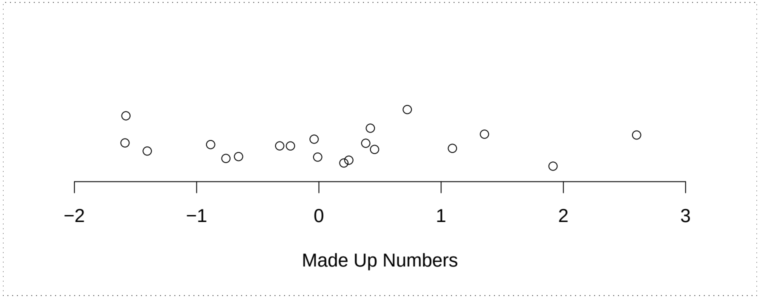
- If there are more than just a few points in a one dimensional scatterplot, there is a high chance that some of the points will be (close to) coincident.
- This can make it difficult to judge the number of points lying in different regions of the plot.
- The solution to this problem is to spread the points at right angles to axis they are being plotted on.
- This is known as *jittering*.

Random Jittering

The computationally simplest form of jiitering is obtained by using a jittered value which is uniformly distributed.

```
> y = runif(length(x), -.75, .75)
> par(mar = c(5.1, 2.1, 4.1, 2.1))
> plot.new()
> plot.window(xlim = range(pretty(x)),
              ylim = c(-1, 1))
> points(x, y)
> axis(1)
> title(xlab = "Made Up Numbers")
```

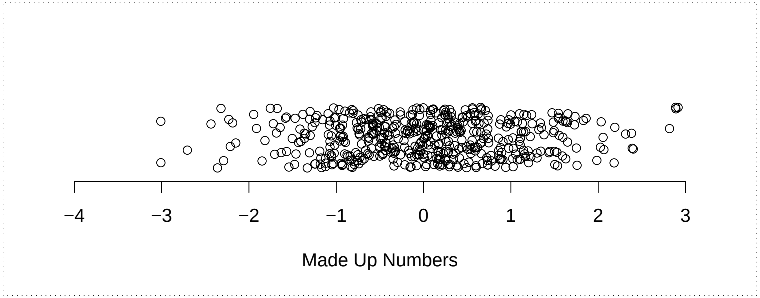
Jittering (20 Points)



Visual Artifacts

- Independent sampling from a uniform distribution does not produce a set of outcomes which are uniformly spread over the range of the distribution.
- Instead, there are both clusters and gaps in the set of outcomes.
- This kind of non-uniformity can lead to visual artifacts in jittered plots.

Vertical Jittering (500 Points)

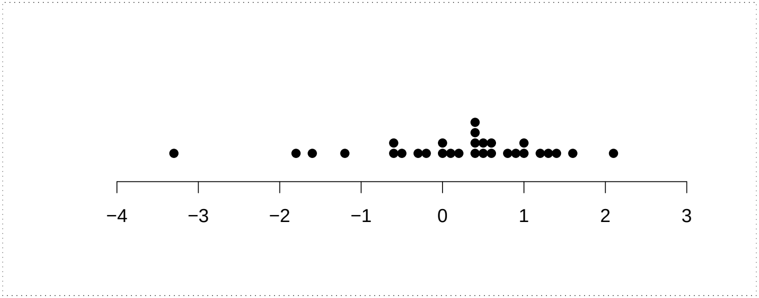


Point Stacking

In very granular data, overplotting will produce points which exactly overlay each other. It can be useful to simply stack equal values beside each other.

```
> x = round(rnorm(30), 1)
> f = function(x) 1:length(x) - 1
> o = unlist(lapply(split(x, factor(x)), f),
             use.names = FALSE)
> y = yinch(par("cin")[2]) * (o - max(o)/2)
> plot.new()
> plot.window(xlim = range(pretty(x)),
              ylim = c(-1, 1))
> points(sort(x), y, pch = 19)
> axis(1)
```

A Stacked 1D Scatterplot (20 Points)



Texturing

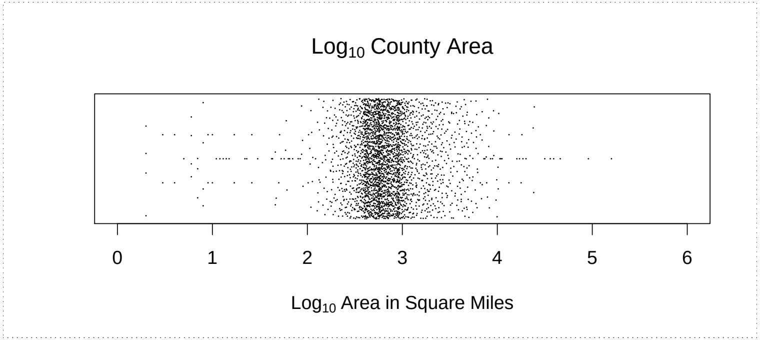
- There is another jittering technique which combines aspects of both random and deterministic jittering.
- The technique is sadly neglected because it is described in a very obscure source.

  Tukey, J. and Tukey P. (1990).
  "Strips Displaying Empirical Distributions I. Textured Dot Strips," Bellcore Technical Memorandum.

- The technique is similar in spirit to random jittering but avoids the visual artifacts that technique produces.

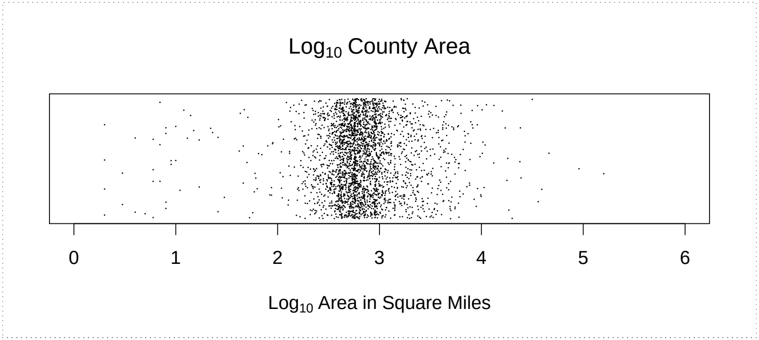
County Areas: Textured Stripplot

There are 2789 counties of varying size in the United States.



County Areas: Jittering

There are 2789 counties of varying size in the United States.



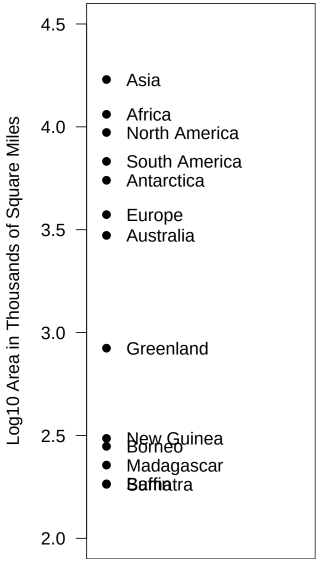
Dealing With Labels

- Often there is additional important information associated with values we are plotting.
- Most commonly this information is in the form of a label which identifies the observation.
- A simple way of plotting points and labels is to plot the points vertically with the labels offset to one side.

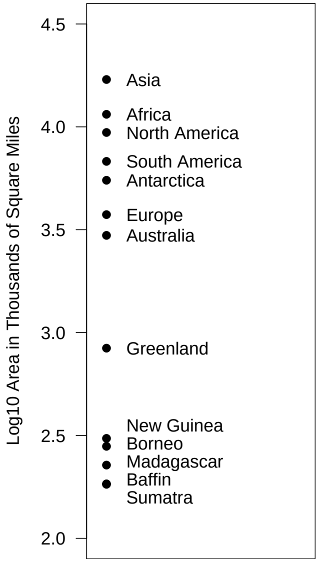
Labelling Points

```
> par(mar = c(0, 4.2, 0, 0))
> x = rep(0, length(land))
> plot.new()
> plot.window(xlim = c(0, 1),
              ylim = range(pretty(land)),
              xaxs = "i")
> points(x + xinch(0.25), land, pch = 19)
> text(x + xinch(0.5),
       land, names(land), adj = c(0, .5))
> axis(2, las = 1)
> title(ylab =
        "Log10 Area in Thousands of Square Miles")
> box()
```

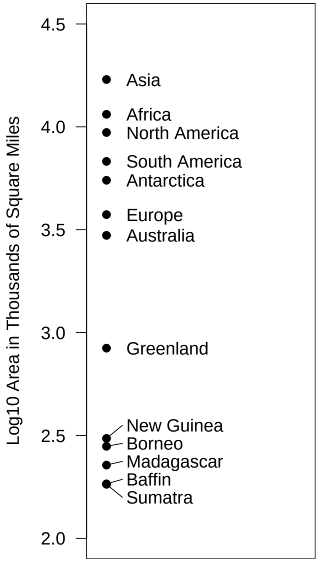
Sizes of the Earth’s Largest Landmasses



Sizes of the Earth’s Largest Landmasses



Sizes of the Earth’s Largest Landmasses



Labelling Points

```
> par(mar = c(0, 4.2, 0, 0))
> x = rep(0, length(land))
> plot.new()
> plot.window(xlim = c(0, 1),
              ylim = range(pretty(land)),
              xaxs = "i")
> points(x + xinch(0.25), land)
> text(x + xinch(0.5),
       separate(land, .85 * yinch(par("cin")[2])),
       names(land), adj = c(0, .5))
> axis(2, las = 1)
> title(ylab =
       "Log10 Area in Thousands of Square Miles")
> box()
```

Label Separation

- The call `separate(values, delta)` takes a set of locations in `values` and returns an adjusted set of locations which are no closer than `delta` together.
- The value of `delta` is on the scale of `values`.
- The height of a character string, in inches, can be computed with the call `par("cin")[2]`.
- This can be converted by y axis data units with the function `yinch()`.

The Separation Algorithm

The algorithm works by merging observations which are close to each other into groups and spacing the observations within groups at a distance of `delta` from each other.

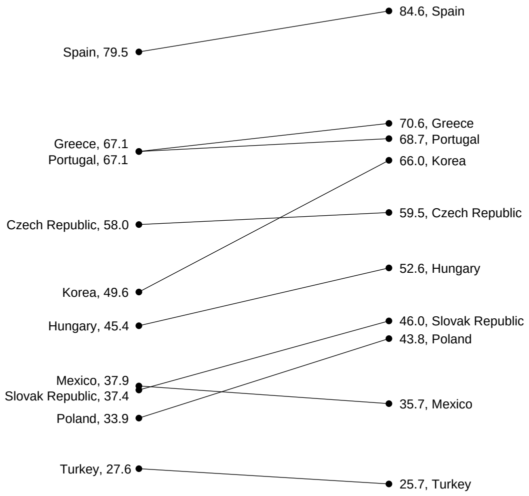
The algorithm is as follows:

1. Sort the observations.
2. Put each observation into its own group.
3. Compute the group min and max.
4. Merge groups which overlap (hard).
5. If groups were merged go to 3.

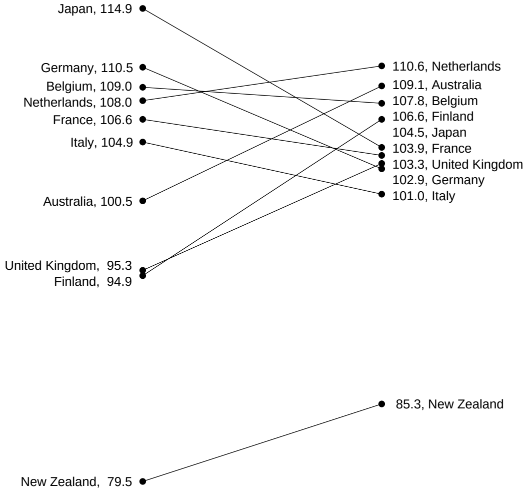
“Before” and “After” Comparisons

By putting labelled one-dimensional displays next to each other it is possible to get a very effective display for paired observations.

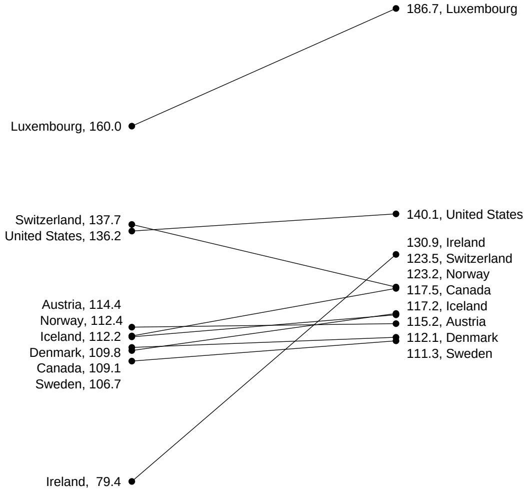
Relative Per Capita GDP in 1992 and 2002



Relative Per Capita GDP in 1992 and 2002



Relative Per Capita GDP in 1992 and 2002


186.7, Luxembourg
Luxembourg, 160.0
140.1, United States
Switzerland, 137.7
United States, 136.2
130.9, Ireland
123.5, Switzerland
123.2, Norway
117.5, Canada
Austria, 114.4
117.2, Iceland
Norway, 112.4
115.2, Austria
Iceland, 112.2
112.1, Denmark
Denmark, 109.8
111.3, Sweden
Canada, 109.1
Sweden, 106.7
Ireland, 79.4