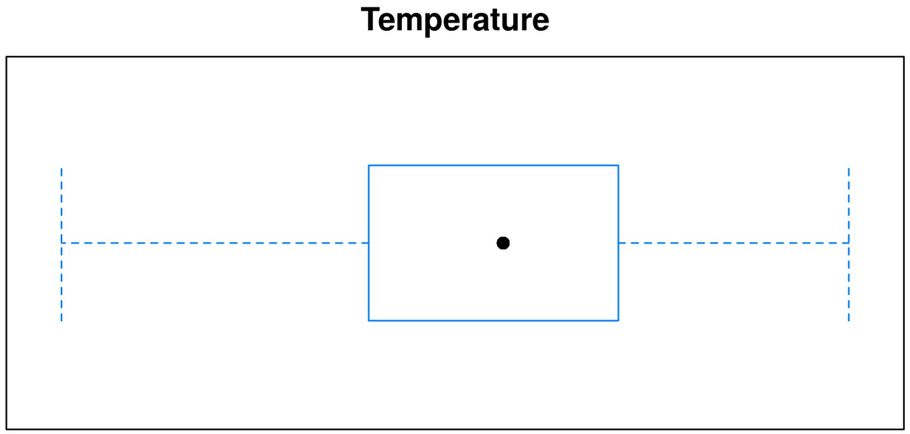
DATA VISUALIZATION IN R WITH LATTICE

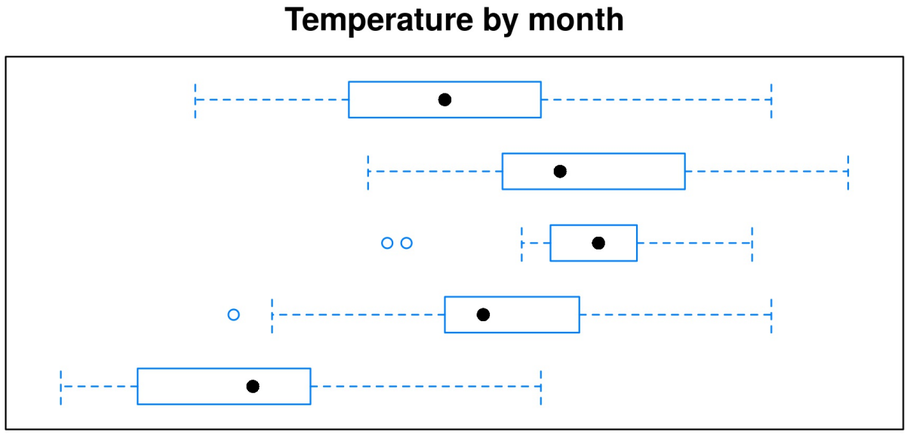
Conditioning

Deepayan Sarkar
Associate Professor, Indian Statistical Institute

Comparison

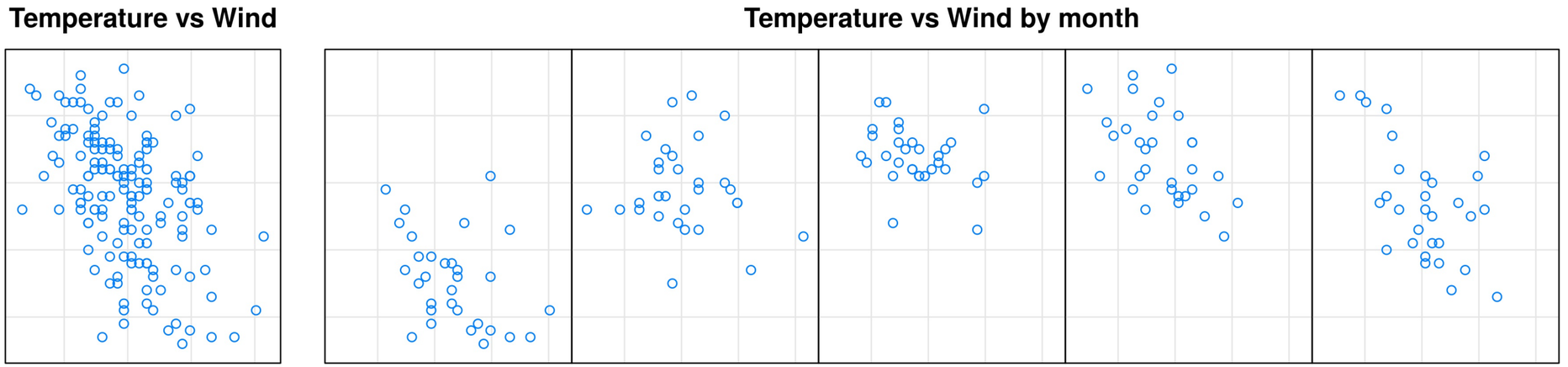
- Goal: identify sources of variability
- Compare different subgroups of data





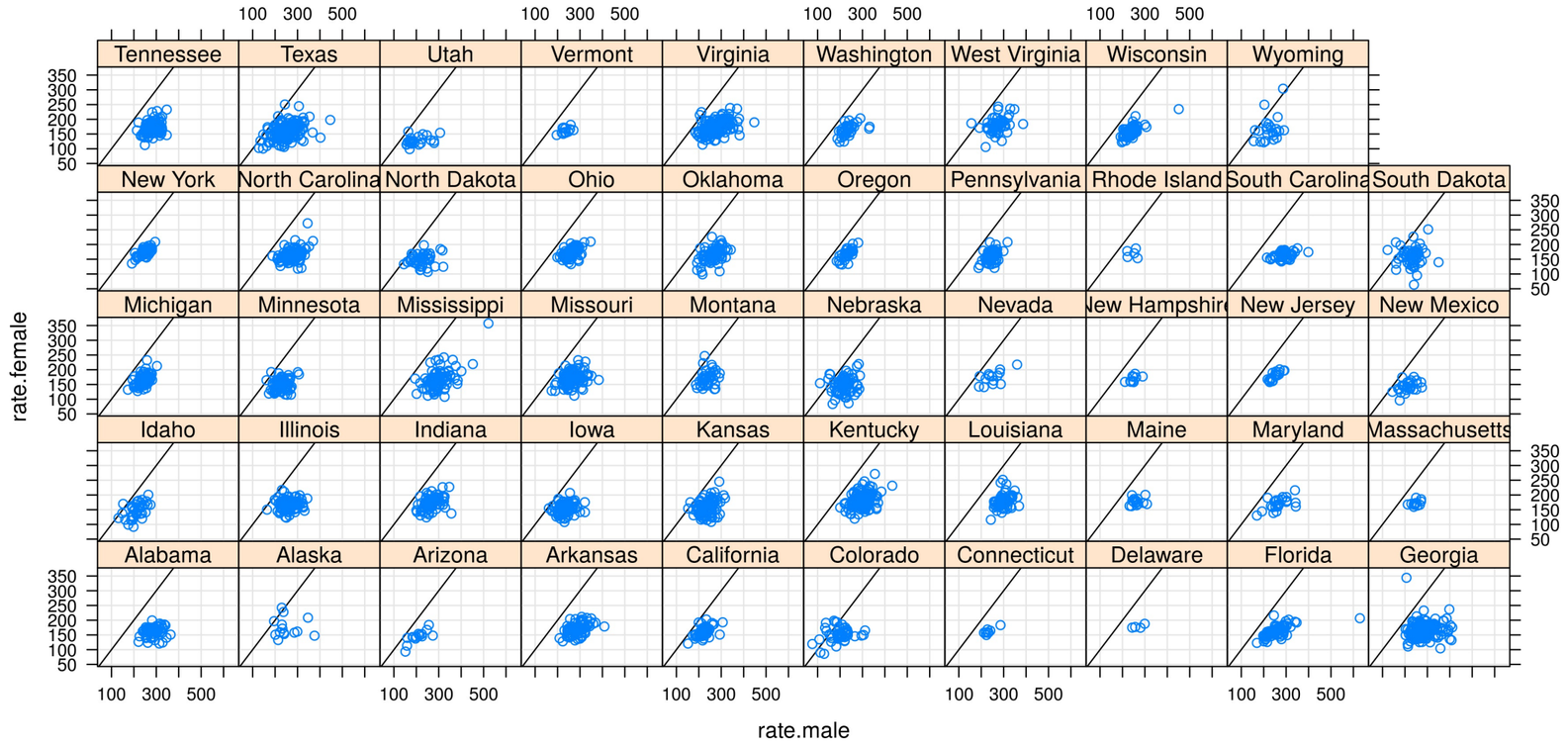
Small multiples (conditioning / faceting)

- Goal: identify sources of variability
- Comparing different subgroups of data



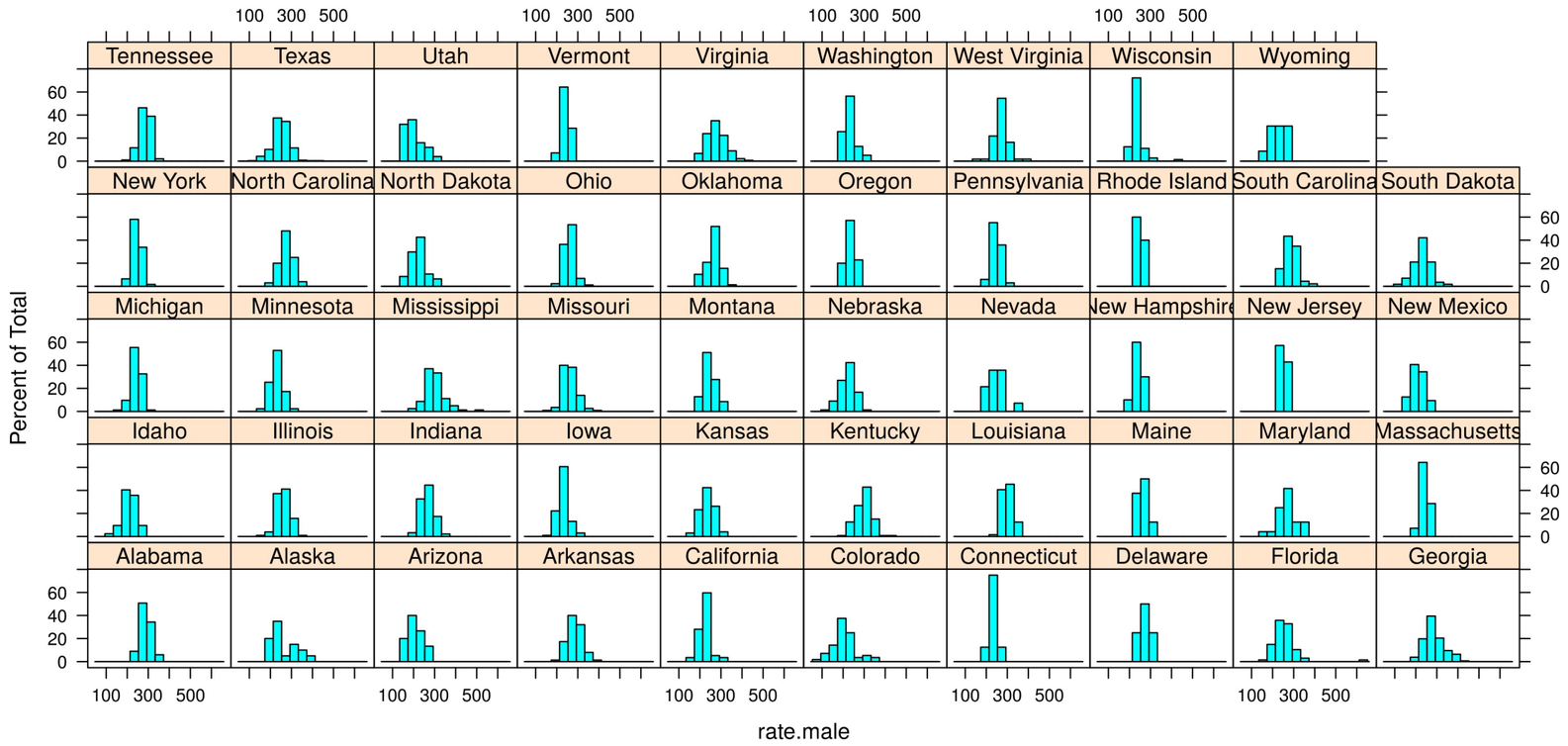
Conditioning in lattice

```
> library(lattice)
> data(USCancerRates, package = "latticeExtra")
> xyplot(rate.female ~ rate.male | state, data = USCancerRates,
+        grid = TRUE, abline = c(0, 1))
```



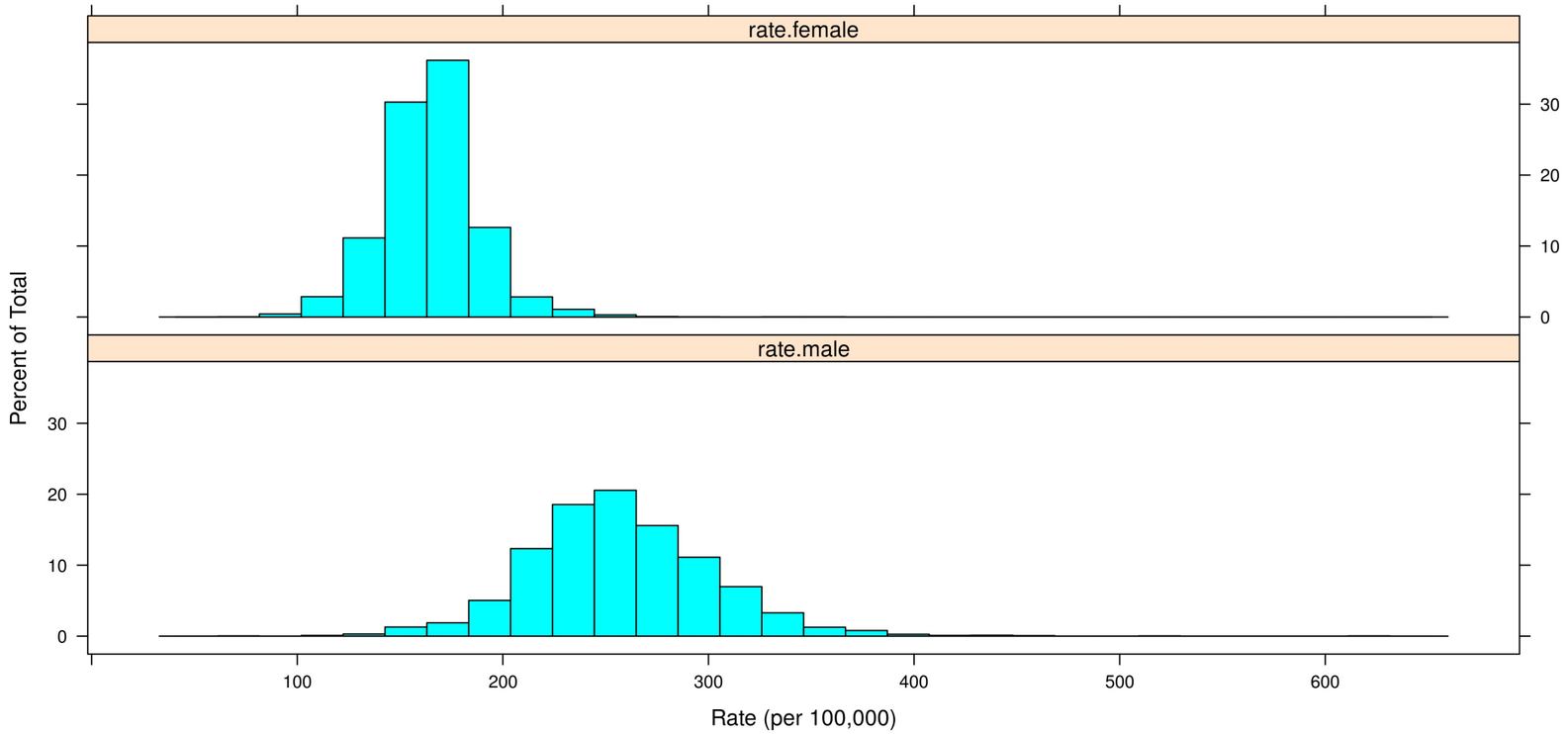
Another conditioned plot

```
> histogram( ~ rate.male | state, data = USCancerRates, nint = 15)
```



Another way to condition

```
> histogram( ~ rate.male + rate.female, data = USCancerRates,
+            nint = 30, outer = TRUE, layout = c(1, 2),
+            xlab = "Rate (per 100,000)")
```



New arguments

```
> histogram( ~ rate.male + rate.female, data = USCancerRates,
+           nint = 30, outer = TRUE, layout = c(1, 2),
+           xlab = "Rate (per 100,000)")
```

- outer = TRUE | FALSE: controls how variables separated by + are interpreted. More details in exercises.

- layout = c(ncol, nrow, npages): controls arrangement of panels in a matrix-type layout.

Summary: Conditioning using formula

- Explicit: `y ~ x | g` or `~ x | g`
- More than one conditioning variable also allowed: `~ x | g1 + g2`
- Implicit: `~ x1 + x2`

DATA VISUALIZATION IN R WITH LATTICE

Let's practice!

DATA VISUALIZATION IN R WITH LATTICE

Data summary and transformation, grouping

Deepayan Sarkar
Associate Professor, Indian Statistical Institute

Data summary

- Summarizing or transforming data can be useful especially for reporting
- We have seen plots of death rates per county
- Can also summarize to get rates per state
- Useful function: tapply(X, INDEX, FUN, ...)
  - Divide numeric data X for each unique value of INDEX
  - Apply function FUN for each subset of X

Data summary

```
> USCancerRates.state <-
+   with(USCancerRates, {
+     rmale <- tapply(rate.male, state, median, na.rm = TRUE)
+     rfemale <- tapply(rate.female, state, median, na.rm = TRUE)
+     data.frame(Rate = c(rmale, rfemale),
+                State = rep(names(rmale), 2),
+                Gender = rep(c("Male", "Female"),
+                             each = length(rmale)))
+  })
> library(dplyr)
> USCancerRates.state <-
+   mutate(USCancerRates.state, State = reorder(State, Rate))
```

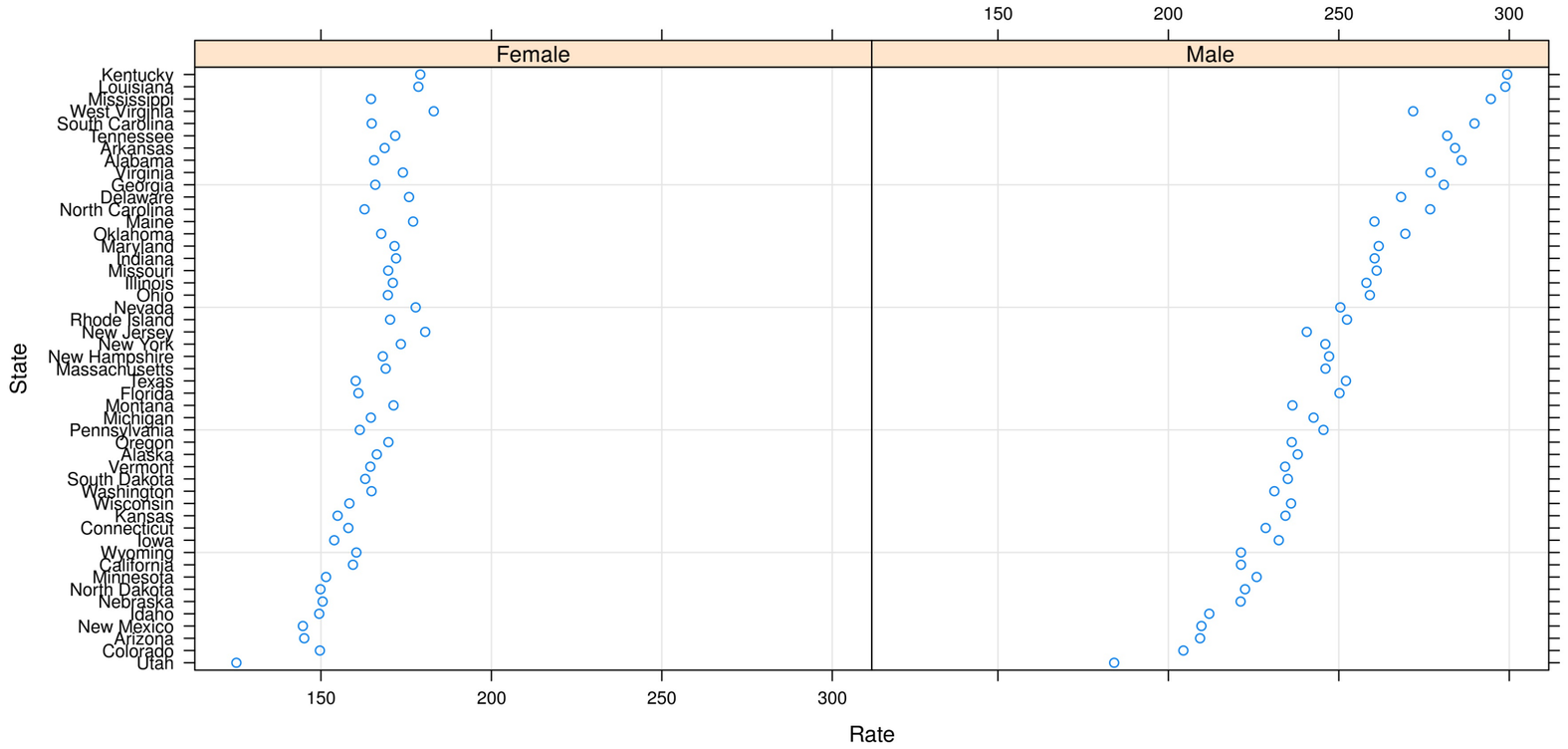
Data summary

```
> USCancerRates.state
```

```
     Rate          State Gender
1  286.00        Alabama   Male
2  237.95         Alaska   Male
3  209.30        Arizona   Male
4  284.10       Arkansas   Male
5  221.30     California   Male
6  204.40       Colorado   Male
7  228.55    Connecticut   Male
8  268.25       Delaware   Male
9  250.20        Florida   Male
10 280.80        Georgia   Male
...
90 171.80      Tennessee Female
91 160.20          Texas Female
92 125.20           Utah Female
93 164.50        Vermont Female
94 174.05       Virginia Female
95 164.85     Washington Female
96 183.10  West Virginia Female
97 158.35      Wisconsin Female
98 160.40        Wyoming Female
```

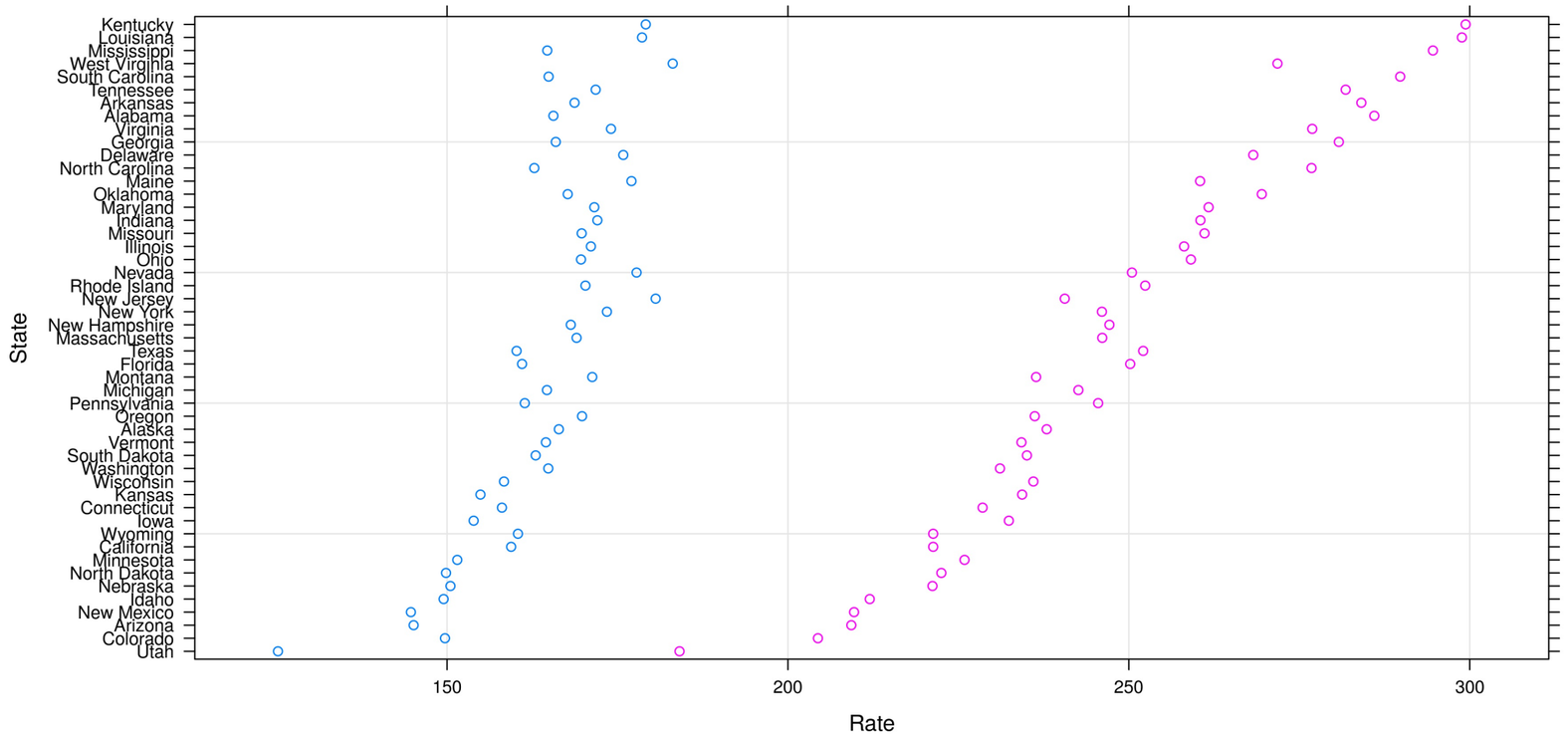
Plotting summary data

```
> xyplot(State ~ Rate | Gender, USCancerRates.state, grid = TRUE)
```



Plotting summary data

```
> xyplot(State ~ Rate, USCancerRates.state, grid = TRUE, groups = Gender)
```



New concept: groups

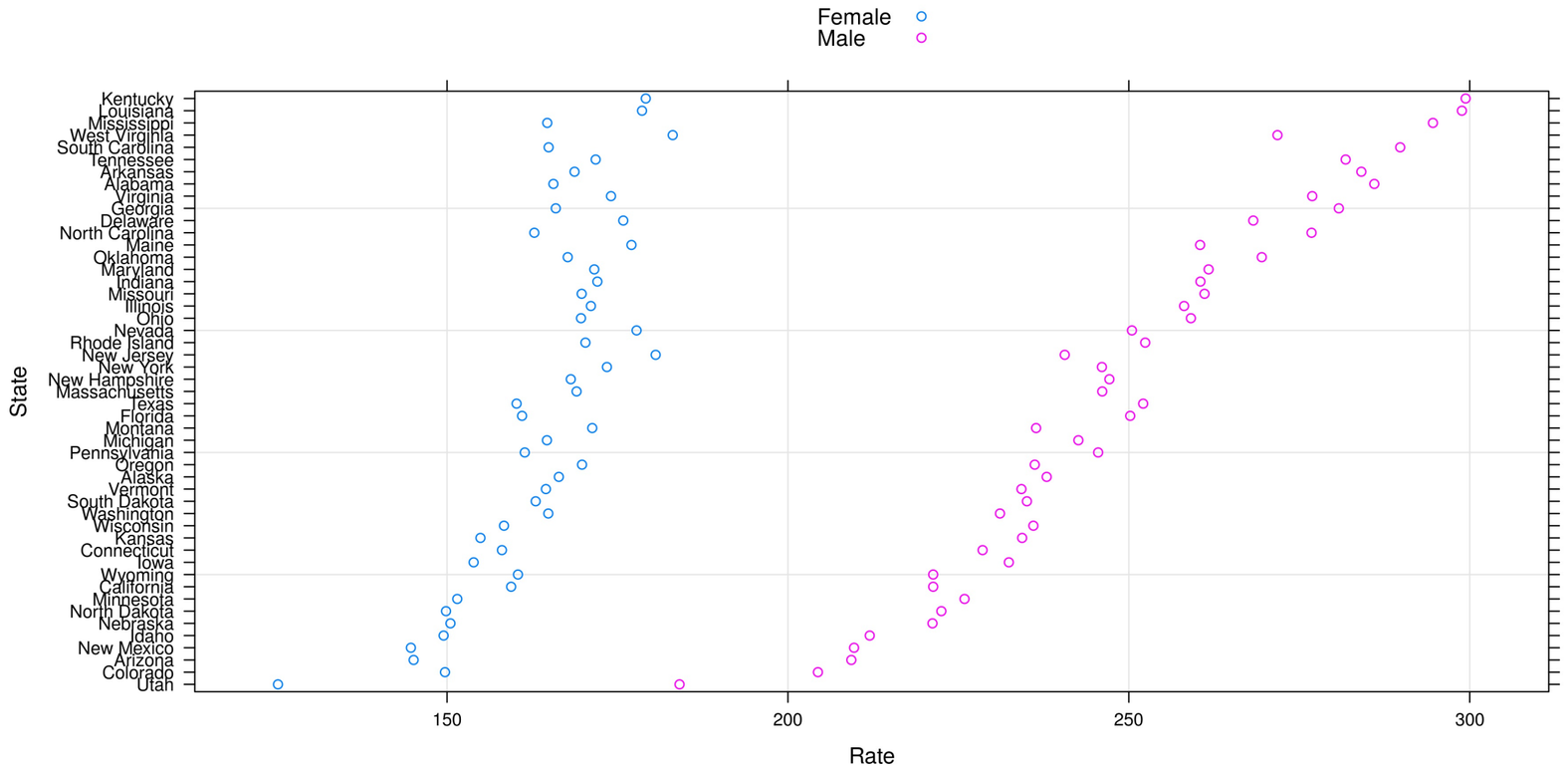
- New argument: groups
- Defines subgroups plotted within same panel
- Differentiated using color (or other graphical parameters)
- Expression evaluated in data

Groups and legends

- Groups are shown using different colors
- Useful to have legend matching colors and groups
- Not present by default
- Use `auto.key = TRUE`

Groups and legends

```
> xyplot(State ~ Rate, USCancerRates.state, grid = TRUE,
+        groups = Gender, auto.key = TRUE)
```



DATA VISUALIZATION IN R WITH LATTICE

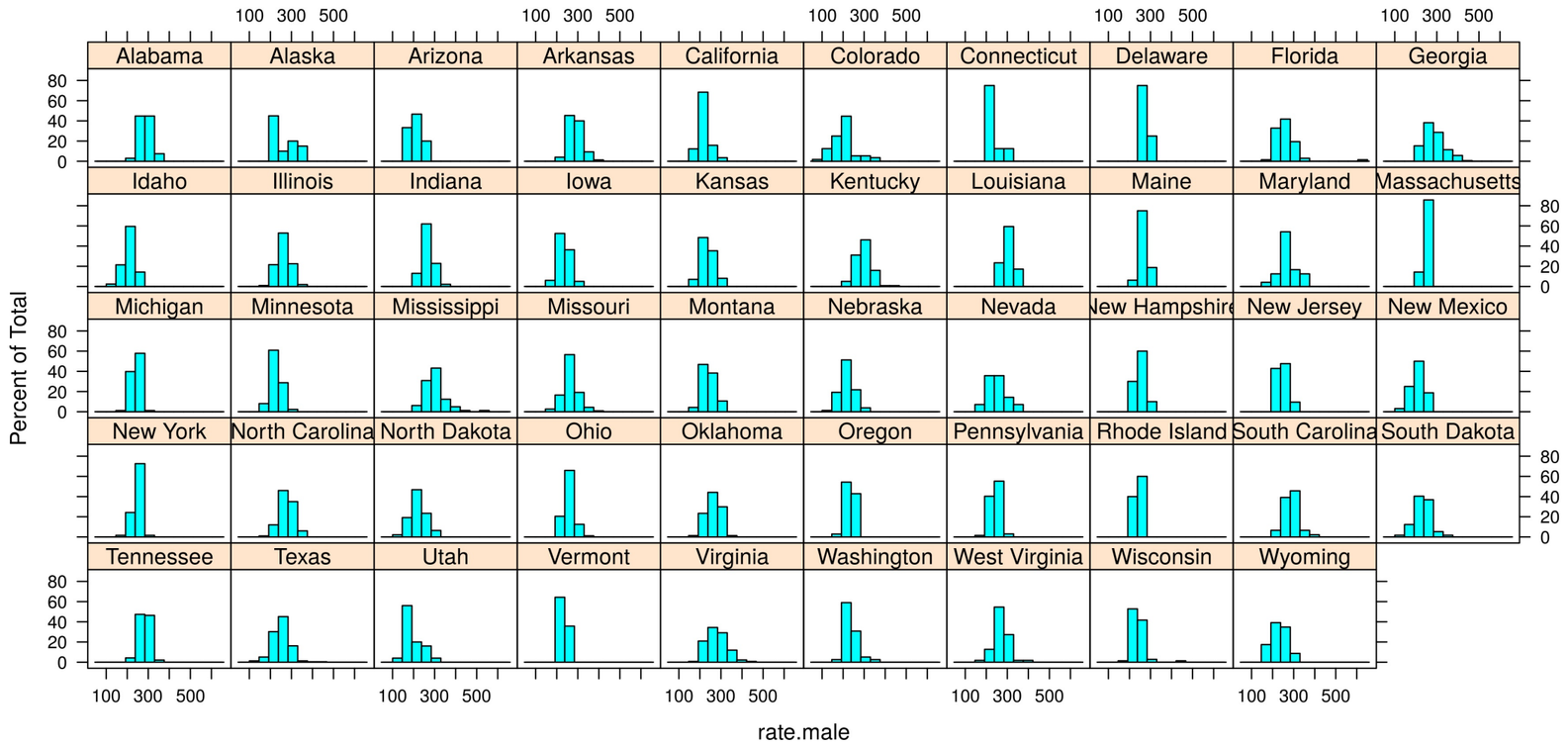
Let's practice!

DATA VISUALIZATION IN R WITH LATTICE

Incorporating external data sources

Deepayan Sarkar
Associate Professor, Indian Statistical Institute

Comparison difficult with too many panels



Possible solutions

- Divide up the panels into multiple pages
- Aggregate panels into fewer groups

Aggregating states

- Premise: neighbouring states should show similar behavior
- Can use our own grouping, or use some pre-defined grouping

Example: state.division

Use the built-in dataset state.division

```
> data.frame(state.name, state.division)
```

```
       state.name     state.division
1         Alabama East South Central
2          Alaska            Pacific
3         Arizona           Mountain
4        Arkansas West South Central
5      California            Pacific
6        Colorado           Mountain
7     Connecticut        New England
8        Delaware     South Atlantic
9         Florida     South Atlantic
10        Georgia     South Atlantic
11         Hawaii            Pacific
12          Idaho           Mountain
13       Illinois East North Central
14        Indiana East North Central
15           Iowa West North Central
...
```

Adding division for each county

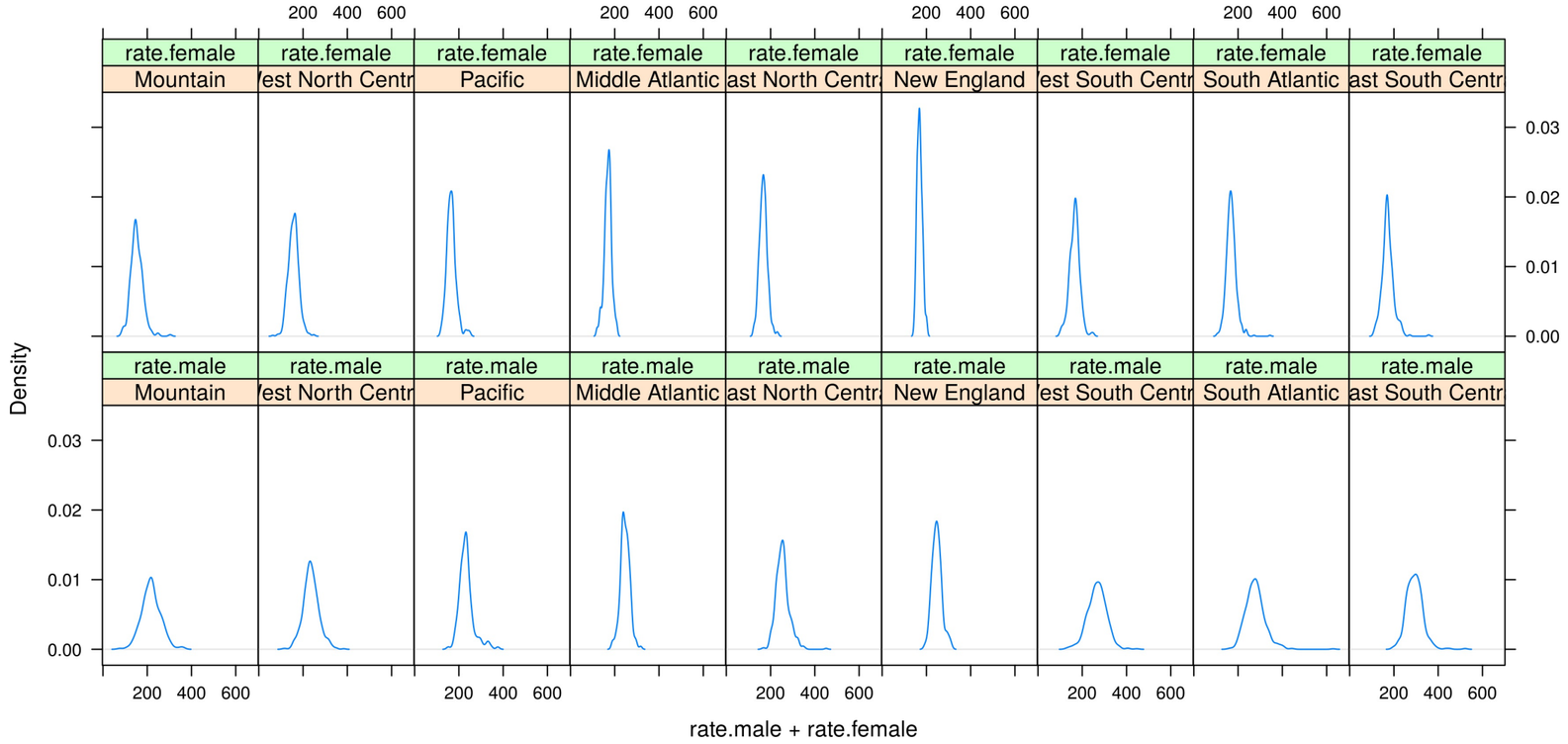
```
> index <- match(USCancerRates$state, state.name)
> USCancerRates$division <- state.division[index]
> summary(USCancerRates$division)
```

```
       New England    Middle Atlantic     South Atlantic
                67                150                587
East South Central West South Central East North Central
               362                451                437
West North Central           Mountain            Pacific
               582                254                151
```

```
> library(dplyr)
> USCancerRates <-
+     mutate(USCancerRates,
+          division.ordered = reorder(division, rate.male + rate.female,
+                                     median, na.rm = TRUE))
```

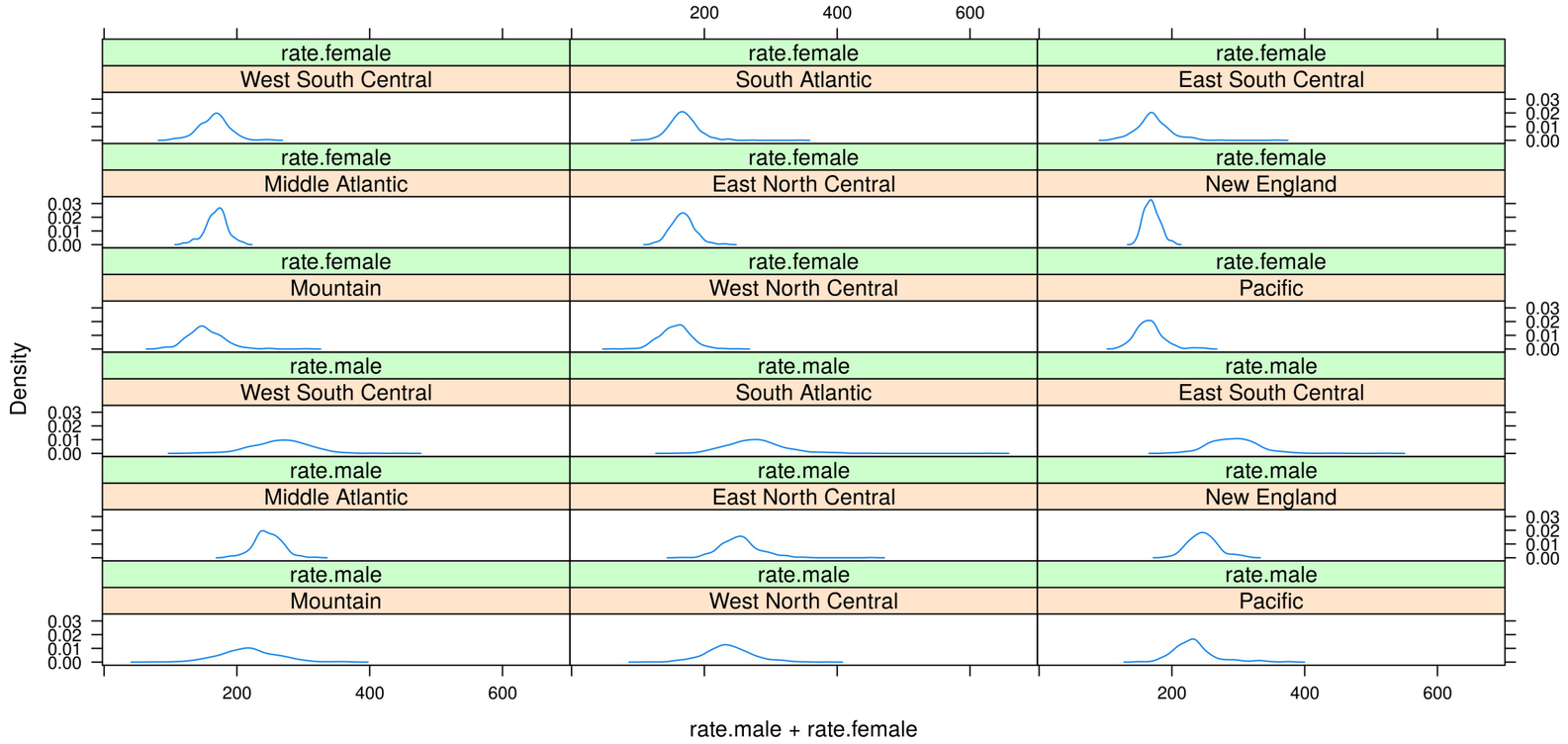
Using the new variable

```
> densityplot(~ rate.male + rate.female | division.ordered, USCancerRates,
+             outer = TRUE, plot.points = FALSE, ref = TRUE)
```



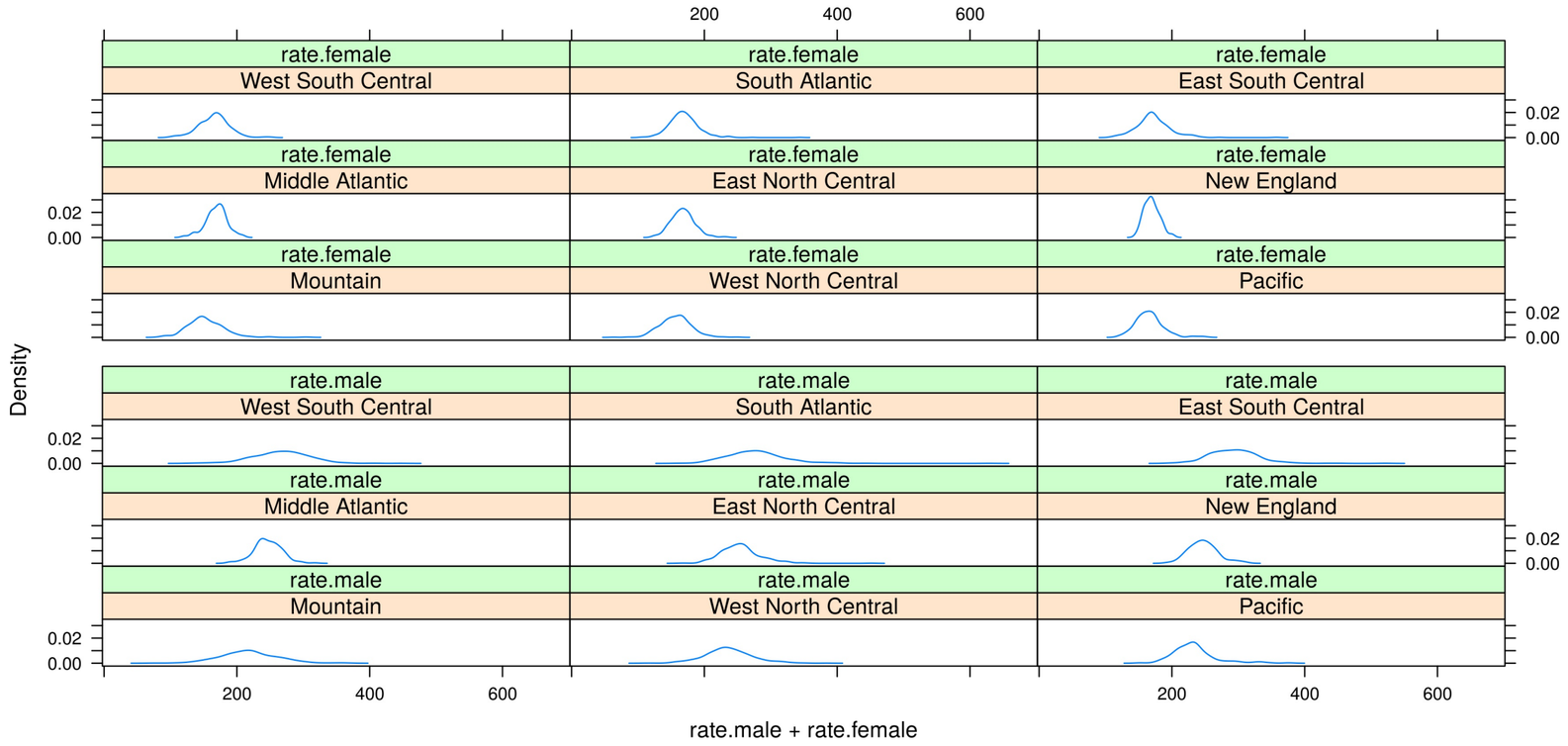
A better layout

```
> densityplot(~ rate.male + rate.female | division.ordered, USCancerRates,
+             outer = TRUE, plot.points = FALSE, layout = c(3, 6))
```



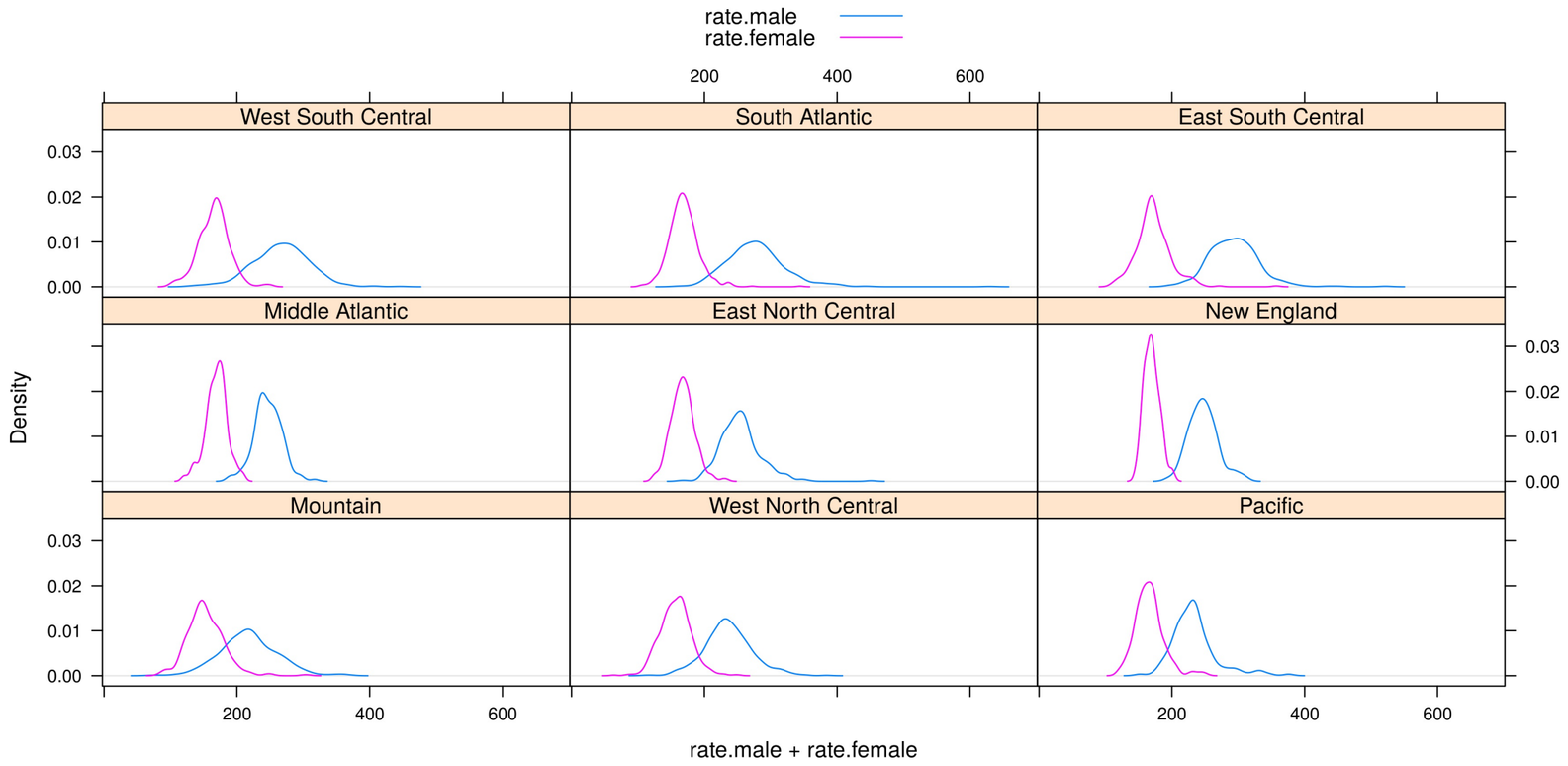
Adding separation between rows and columns

```
> densityplot(~ rate.male + rate.female | division.ordered, USCancerRates,
+             outer = TRUE, plot.points = FALSE, layout = c(3, 6),
+             between = list(y = c(0,0,1,0,0)))
```



Grouping instead of conditioning

```
> densityplot(~ rate.male + rate.female | division.ordered, USCancerRates,
+             outer = FALSE, plot.points = FALSE, ref = TRUE,
+             layout = c(3, 3), auto.key = TRUE)
```



DATA VISUALIZATION IN R WITH LATTICE

Let's practice!

DATA VISUALIZATION IN R WITH LATTICE

The "trellis" object

Deepayan Sarkar
Associate Professor, Indian Statistical Institute

Plots as objects

```
> # Create trellis object
> tplot <-
+     densityplot(~ rate.male + rate.female | division.ordered,
+                 data = USCancerRates, outer = TRUE,
+                 plot.points = FALSE, as.table = TRUE)
```

```
> class(tplot)
```

```
[1] "trellis"
```

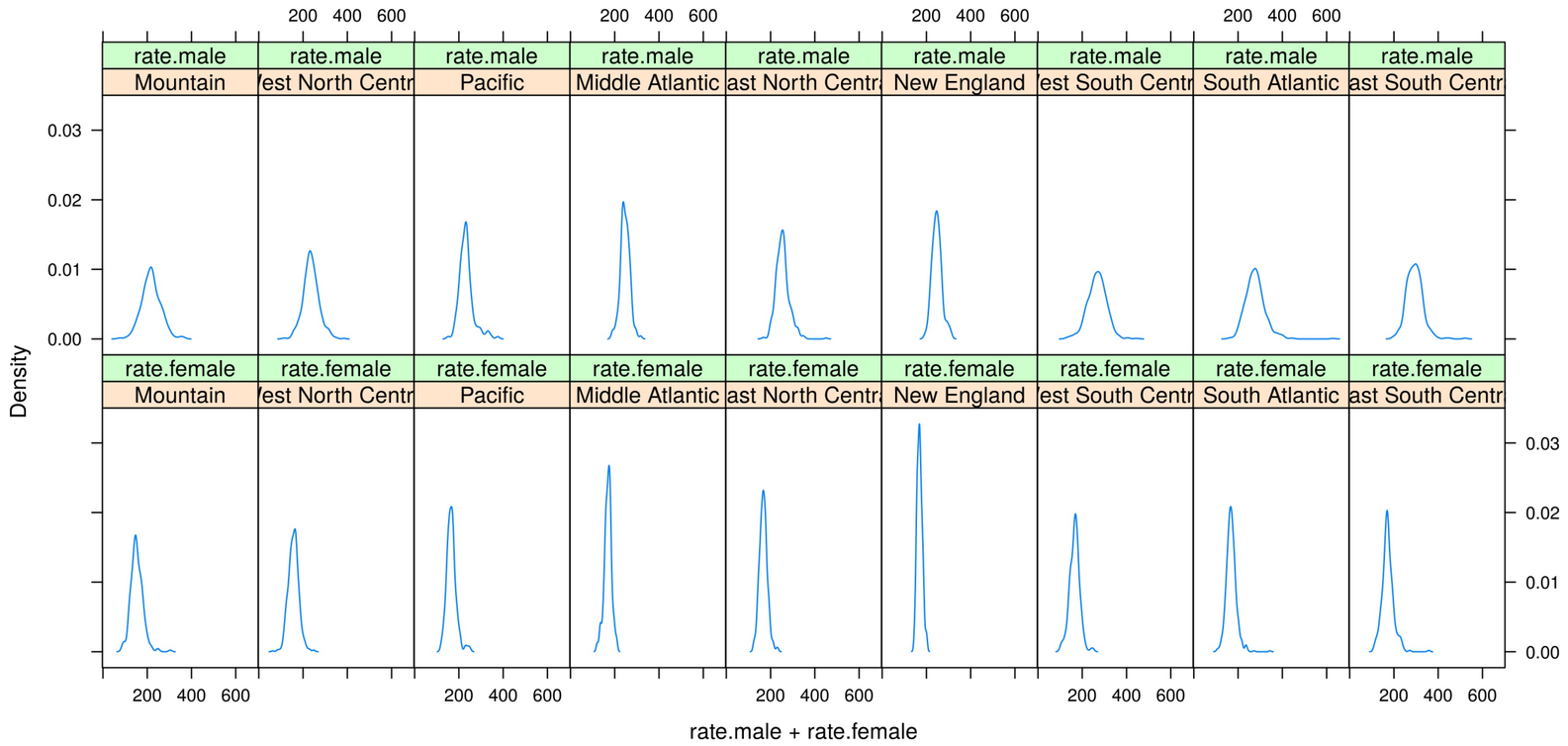
Summary of "trellis" objects

```
> summary(tplot)
```

```
Call:
densityplot(~rate.male + rate.female | division.ordered, data = USCancerRates,
    outer = TRUE, plot.points = FALSE, as.table = TRUE)

Number of observations:

division.ordered     rate.male rate.female
  Mountain                 254         254
  West North Central       582         582
  Pacific                  151         151
  Middle Atlantic          150         150
  East North Central       437         437
  New England               67          67
  West South Central       451         451
  South Atlantic           587         587
  East South Central       362         362
```

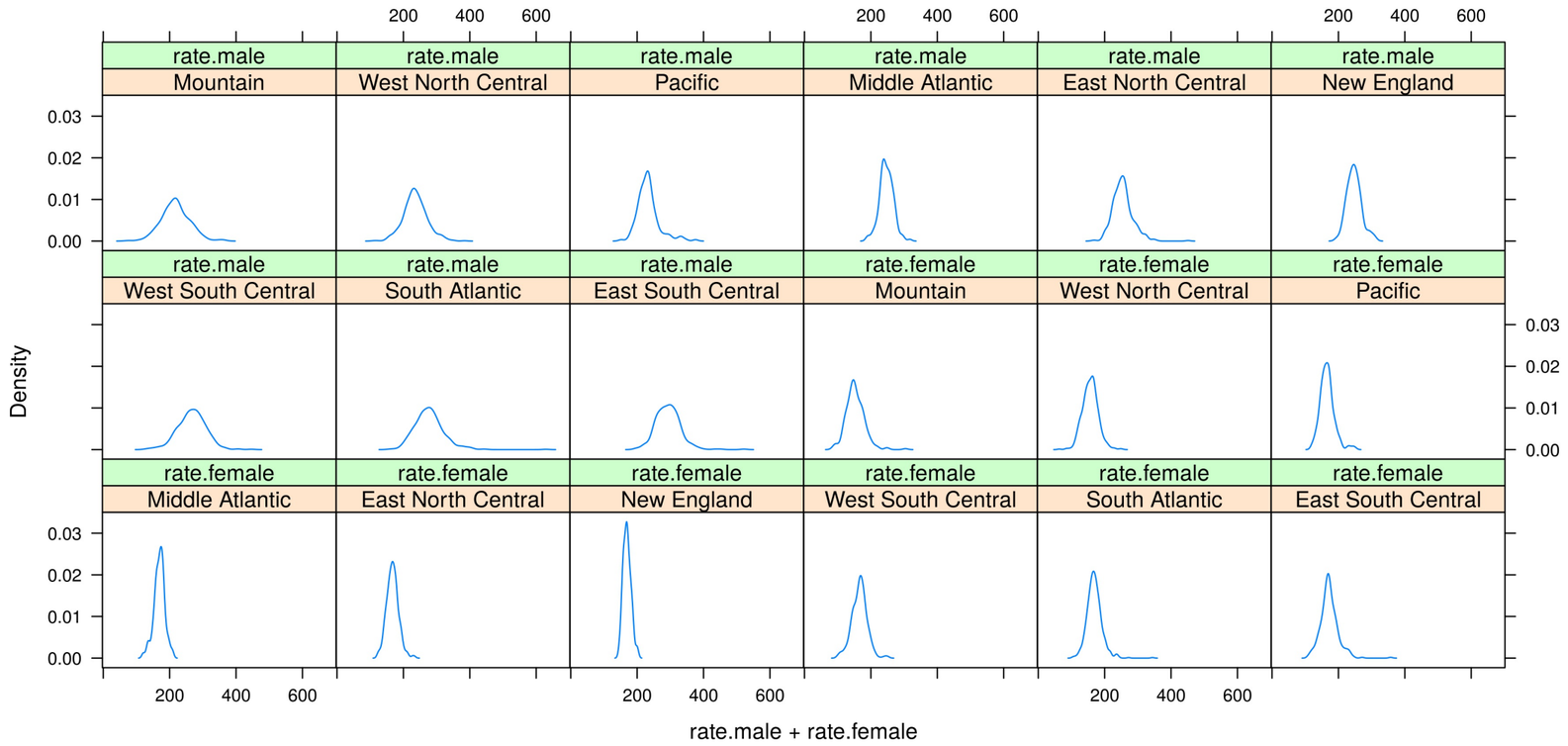
Drawing "trellis" objects

```
> tplot
```



Updating "trellis" objects

```
> update(tplot, layout = c(6, 3))
```



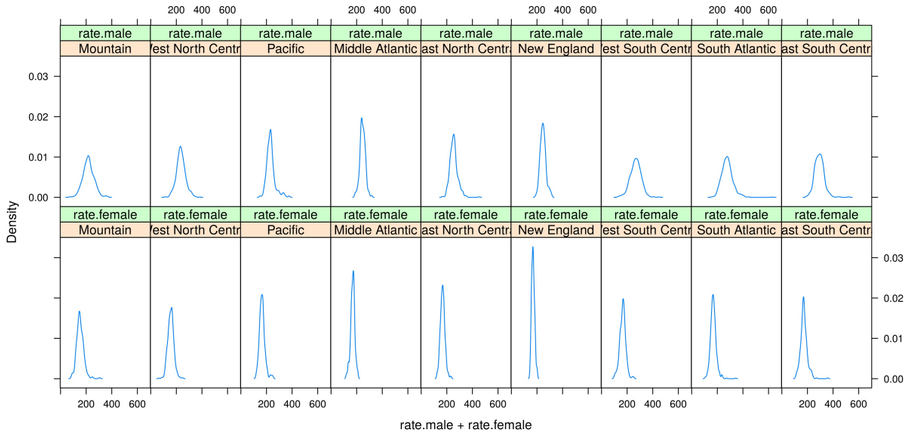
"trellis" objects as arrays

```
> dim(tplot)
[1] 9 2

> dimnames(tplot)
$division.ordered
[1] "Mountain"           "West North Central"
[3] "Pacific"            "Middle Atlantic"
[5] "East North Central" "New England"
[7] "West South Central" "South Atlantic"
[9] "East South Central"

[[2]]
[1] "rate.male"   "rate.female"
```

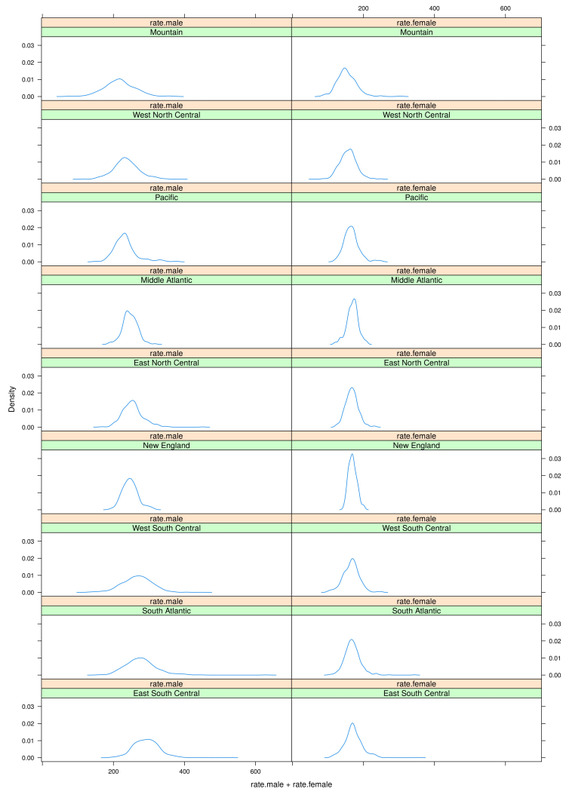
"trellis" objects as arrays

```
> # Transpose tplot like a matrix
> t(tplot)
```

Original tplot



Transposed tplot



DATA VISUALIZATION IN R WITH LATTICE

Let's practice!