Enough R for Data and Computing Fundamentals

Process: Start with the original, immutable data tables. Transform variables, subset cases, group cases, and join tables as needed to create a single table with the variables you want to display or model.

Sources of Data

Data are available through many sources, but a few data tables are regularly used for examples in DCF. You will usually read the data table into R with the `data()`, or from the Internet with `fetchData()`, or `fetchGapminder()` functions:

```
data(nhanes)
data(OrdwayBirdsOrig)
data(WakeVotersSmall)
```

These create data tables with the indicated name.

Assignment & Naming

Use `<-` or `=` to store an object by name. Use short, mnemonic names. You can use assignment to create copies of existing tables or to read in new ones.

```
birds <- OrdwayBirdsOrig
voters <- WakeVotersSmall
```

Know your Data

Be prepared to answer these questions about any data table:

- What constitutes a **case**?
- How many cases are there?
- What are the **variables**?
- What type is each variable?

How many cases?

```
nrow(birds)
[1] 15829
```

Variable names & Renaming

```
names(voters)
[1] "Age"    "party"  "gender"
```

```
voters <- rename(voters,
              c(gender="Sex"))
```

Variable types

Check the variables explicitly to avoid mistakes:
`factor` mean categorical

```
class(voters)
[1] "data.frame"
class(voters$Age)
[1] "integer"
class(voters$party)
[1] "factor"
```

Dirty Data

Sometimes data will surprise you:

```
class(birds$Month)
[1] "factor"
```

You probably thought `Month` would be numeric.

See the Levels

Categorical variables have levels.

```
levels(birds$Month)
 [1] ""      "1"     "10"    "11"
 [5] "12"    "2"     "25"    "3"
 [9] "4"     "5"     "6"     "7"
[13] "8"     "9"     "Month"
```

Someone entered month “25” and the word “Month”.

Simple Data Cleaning

Categorical → quantitative

```
birds <- transform(birds,
  Month=as.numeric(
    as.character(Month)))
```

Change Type of Variables

Quantitative → categorical

```
nhanes <- transform(nhanes,
  cut(age,breaks=c(0,18,65,100),
  labels=c("kid","adult","senior")))
```

Evenly spaced groups

```
nhanes <- transform(nhanes,
           agegps=cut(age,3))
```

Evenly populated groups

```
nhanes <- transform(nhanes,
           wtgps=ntiles(wgt,3))
```

Group Summaries

Specify the variable or variables to use for grouping, and the operations. By default, a count of the number in each group.

```
groupBy(voters,by=party)
  party count
1   DEM  4101
2   REP  3098
3   UNA  2783
```

Median age of voters, by party:

```
groupBy(voters,by=party,
         medage=mean(Age))
  party   medage
1   DEM 46.84638
2   REP 46.43351
3   UNA 40.71398
```

Use > 1 grouping variables.

```
groupBy(voters,
        by=list(party,gender))
```

Subset of Cases

According to a criterion:

```
kids <- subset(nhanes,age<10)
teen <- subset(nhanes,
            age>12 & age<20)
```

Random sample

```
small <- sample(nhanes,size=10)
```

Subset of Variables

```
birds <- subset(OrdwayBirdsOrig,
    select=c("SpeciesName","Month"))
```

Mathematical Ops

```
nhanes <- transform(nhanes,
            area=wst*hgt)
```

Joining Two Tables

Example: Drop the low-count species of birds.

```
counts <- groupBy(birds,
                  by=SpeciesName)
```

	SpeciesName	count
1	Arkansas Kingbird	1.00
2	Bank Swallow	21.00
3	Bay-breasted Warbler	2.00

```
birds <- join(birds,counts)

Joining by:  SpeciesName
```

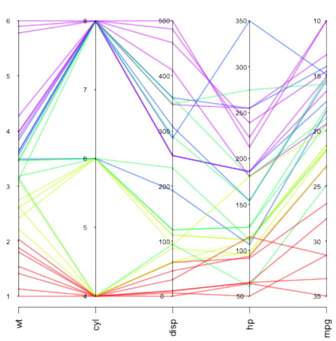
	SpeciesName	Month	count
1	Bank Swallow	6	21.00
2	Bay-breasted Warbler	9	2.00
3	Bank Swallow	7	21.00
4	Bay-breasted Warbler	9	2.00

```
birds <- subset(birds,
                count>200)
```

Choice in `join()`: • Variables to use for matching; • which tables' cases to keep.

Parallel Coordinates

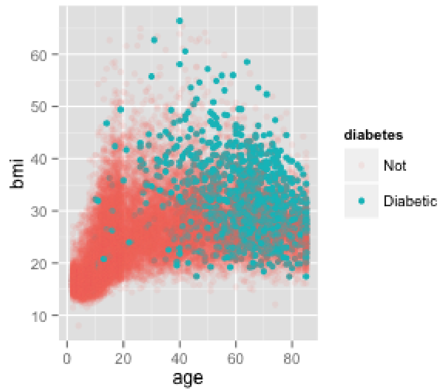
```
parallelPlot(mtcars,
    ~wt + cyl + disp + hp - mpg,
    lwd=5,axes=TRUE)
```



Scatter Plots

To generate interactively:

```
mScatter(nhanes)
```



Required Variables in table:

- x-position: Quant. e.g. `Age`
- y-position: Quant. e.g. `BMI`

Optional variables:

- Size: Quant. or Categorical (not used here)
- Color: Quant. or Categorical, e.g, `diabetes`
- Transluscency `diabetes`

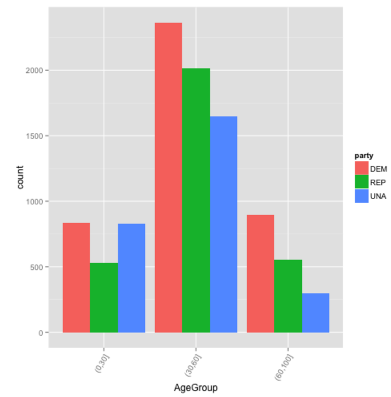
Choices:

- Log axes (not used here)

Bar Plots

Make interactively with

```
mBar(voters)
```



Typically generated from **grouped** data. Required Variables in table:

- x-axis: Categorical e.g. `AgeGroup`
- y-axis: Quant. typically a count.

Optional Variable:

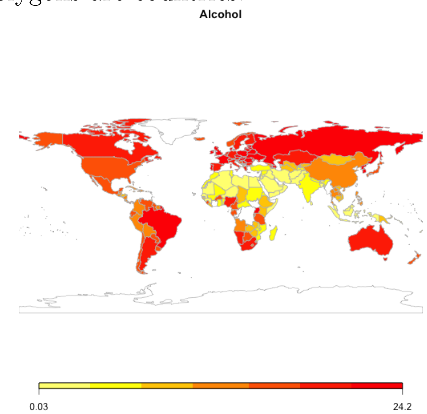
- Sub-divisions of bar groups: Categorical, e.g. `party`

Choices:

- Sub-bar arrangment: stacked, dodged, proportional.

Maps

Choose a program for the geometry of interest. Here, `mWorldMap()`, so the polygons are countries.



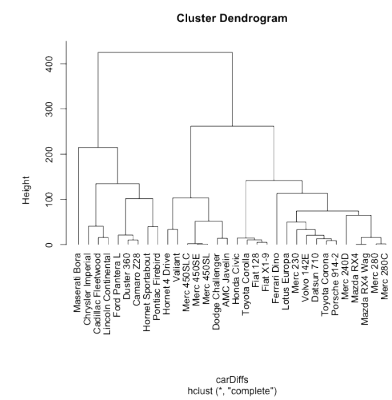
Required Variables in table:

- Polygon: Categorical, here `Country`
- Color: Categ. or Quant.

One case per polygon!

Trees

Find distances between each pair of individual cases.




```
hc <- hclust(dist(mtcars))
plot(hc, hang=-1)
```