September 2014

Enough R for Data & Computing Fundamentals

Daniel Kaplan

Getting Started Load the package whenever you start a new session.

```
library(DCF)
```

Overview The data verbs, summary functions, and transformation functions enable you to transfigure data into a glyph- or analysis-ready form.

The basic syntax:

```
Result <-
  DT %>%
  verb1( [some args] ) %>%
  verb2( [more args] ) %>%
  ... and so on as needed ...
```

- `<-` is the assignment symbol.
- `%>%` is the chaining symbol: take the output of the left expression and make it the input of the right expression.
- Lines that **end** with `<-` or `%>%` identify that the next line continues the expression.

Data Tables are organized into cases and variables. Variables are either quantitative or categorical: numbers or words.

- First example data table: `DT`

```
##      name sex height weight
## 1   Alma   F   1.64     54
## 2 Junior   M   1.82     73
## 3   Gary   M   1.71     64
## 4 Kristy   F   1.75     61
```

`sex` is categorical, `height` and `weight` are quantitative.

- Second example table: `Sports`

```
##    name      sport
## 1 Fred   Football
## 2 Alma Water Polo
## 3 Alma     Hockey
## 4 Gary   Football
```

Quick presentation of data tables

```
str( DT )  summary( DT )
nrow( DT ) names( DT )
head( DT ) tail( DT ) glimpse( DT )
```

Data Verbs take a data table as input and return as output a modified table.

Verb	Task	Argument(s)	Example
`filter()`	Winnow cases	Comparison	`filter(year>2000)`
`mutate()`	Adds vars.	Transformation	`mutate(bmi=weight/height^2)`
`summarise()`	Combines cases	Summary	`summarise(ave=mean(height))`
`select()`	Drops vars.	Var. Names	`select(sex, height)`
`arrange()`	Order cases	Var. Names	`arrange(height)`
Join	Combines tables	Data Table	See **Various Joins**
`group_by()`	Split into groups	Var. Names	`group_by(sex)`

All the examples assume a data table is being chained in, e.g. `DT %>% group_by(sex)`.

Grouping Operations

`group_by()` can be used with several data verbs.

Summarize within each group property

```
DT %>% group_by( sex ) %>%
  summarise(tallest=max(height))
```

Compare each case to a group property

```
DT %>% group_by( sex ) %>%
  mutate( rel=height-mean(height))
```

Choose cases from each group.

```
DT %>% group_by( sex ) %>%
  filter( rank(height)==1 )
```

Various Joins differ mainly in how they deal with unmatched cases.

Cases matched with *all* variables that appear in both tables, just `name` in the example.

- Keep all cases that have a match:

`DT %>% inner_join( Sports )`

```
##   name sex height weight      sport
## 1 Alma   F   1.64     54 Water Polo
## 2 Alma   F   1.64     54     Hockey
## 3 Gary   M   1.71     64   Football
```

Note: output has *both* of Alma's sports.

- Keep all cases from left table:

`DT %>% merge( Sports, all.x=TRUE )`

Use `all=TRUE` to keep all cases from both tables.

- Keep unmatched cases:

`DT %>% anti_join( Sports )`

To Use in Arguments to Data Verbs

Summary Functions take a variable as input and return a single number.

```
mean( height, na.rm=TRUE )
max( weight )  n()
min( weight )  n_distinct()
```

Transformation Functions, used with `mutate()`, take one or more variables as input and return a variable (with the same number of cases).

```
rank( var )
pmin( var1,var2) #smaller of the two
var1/(var1+var2) #division, addition
```

Comparison Expressions

`filter()` uses one or more comparison expression to determine which cases to pass through.

```
filter( DT, height < 1.8 )
filter( DT, name=="Junior" )
filter( DT, sex=="F", height < 1.8 )
filter( DT, count>2000, count<10000 )
filter( DT, name%in%c("Alma","Gary"))
```

Variable Names

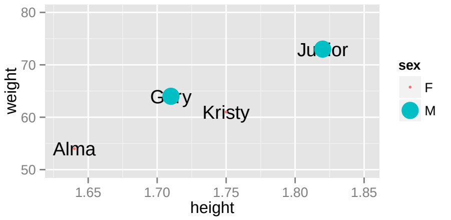
`group_by()`, `select()`, and `arrange()` take one or more variable names as arguments, in addition to the chained in data table.

Graphics with ggplot

- Create a new graphic: `ggplot()`
- Functions to add graphical layers `geom_point()` `geom_text()` `geom_bar()`, etc. Others: `xlab()`, `ylab`, `xlim(low,high)`, `ylim(low,high)`
- Distinguish groups using color `group=sex`
- `aes()` to map variables to graphical attributes (aesthetics).

Example:

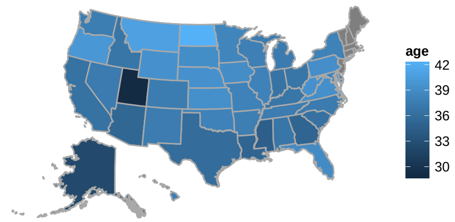
```
DT %>%
 ggplot(aes(x=height,y=weight)) +
 geom_text( aes(label=name) ) +
 geom_point(aes(color=sex,size=sex))
```



Choropleth Maps

`mUSMap()` has a `key=` argument identifies the variable naming the geographic entity. `fill=` specifies the quantity to be plotted.

```
plot(mUSMap(data=StateData,
              key="State",fill="age") )
```



`mWorldMap()` is used in the same way.