Stat405

Groupwise operations

Hadley Wickham

1. Group wise operations

2. Challenges

Projects

Team debrief

- What worked well? What didn't work well?
- What will you change next time?
- Write one page (per team)

Teams

- Unless all group members unanimously agree, teams will be dissolved.
- Fill out http://bit.ly/mOAyFM by 5pm Monday
- If you decide to go, you can say who you absolutely don’t want to work with again and we will respect your wishes.

Group-wise operations

```
# Getting started
library(plyr)
library(stringr)
options(stringsAsFactors = FALSE)

bnames <- read.csv("bnames2.csv.bz2")
births <- read.csv(
  "http://stat405.had.co.nz/data/births.csv")

bnames2 <- join(bnames, births, type = "left")
bnames2 <- mutate(bnames2,
  n = round(prop * births),
  first = str_sub(name, 1, 1),
  last = str_sub(name, -1, -1))
```

Number of people

How do we compute the number of people with each name over all years ?
It's pretty easy if you have a single name. (e.g. how many people with your name were born over the entire 128 years)

How would you do it?

```r
# Split
pieces <- split(bnames2, list(bnames$name))

# Apply
results <- vector("list", length(pieces))
for(i in seq_along(pieces)) {
  piece <- pieces[[i]]
  results[[i]] <- summarise(piece,
    name = name[1], n = sum(n))
}

# Combine
result <- do.call("rbind", results)
```

```
# Or equivalent

counts <- ddply(bnames2, "name", summarise,
  n = sum(n))
```

Input data

Way to split up input

2nd argument to `summarise()`

Function to apply to each piece

x	y
a	2
a	4
b	0
b	5
c	5
c	10

Split

x	y
a	2
a	4
b	0
b	5
c	5
c	10

x	y
a	2
a	4

x	y
b	0
b	5

x	y
c	5
c	10

Split Apply

x	y
a	2
a	4
b	0
b	5
c	5
c	10

x	y
a	2
a	4

3

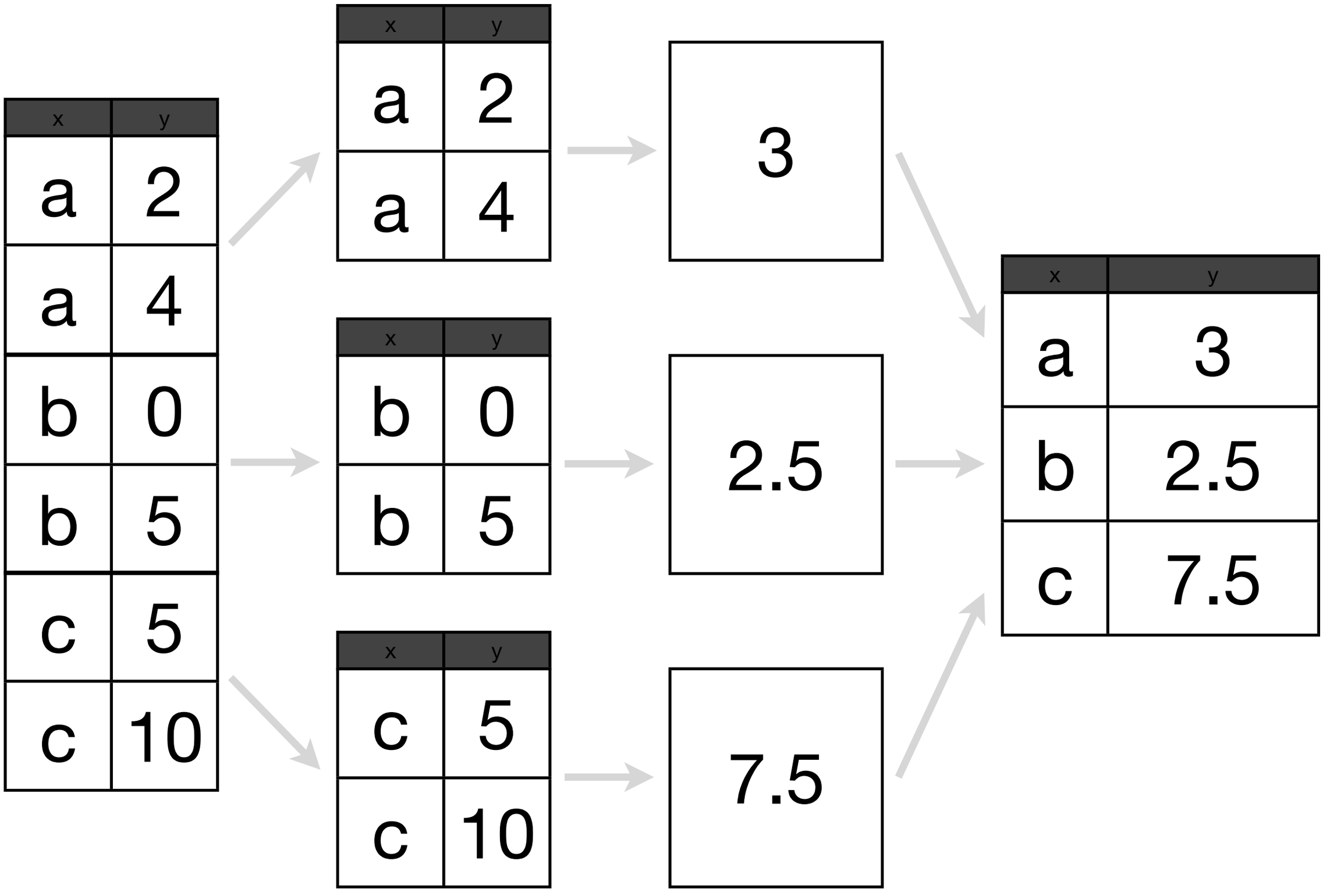
x	y
b	0
b	5

2.5

x	y
c	5
c	10

7.5

Split Apply Combine



Your turn

Repeat the same operation, but use soundex instead of name. What is the most common sound? What name does it correspond to?

```
scounts <- ddply(bnames2, "soundex", summarise,
  n = sum(n))
scounts <- arrange(scounts, desc(n))

# Combine with names
# When there are multiple possible matches,
# join picks the first
scounts <- join(
  scounts, bnames2[, c("soundex", "name")],
  by = "soundex")
head(scounts, 100)

subset(bnames, soundex == "L600")
```

```
scounts <- ddply(bnames2, "soundex", summarise,
  n = sum(n))

# Specialised function for (weighted) counts
# Faster, but only does one thing
scounts <- count(bnames2, "soundex", "n")
```

Transformations

Transformations

What about group-wise **transformations**? e.g. what if we want to compute the rank of a name within a sex and year? (John was the nth most popular boys name in 2008...)

This task is easy if we have a single year & sex, but hard otherwise.

Transformations

What about group-wise **transformations**? e.g. what if we want to compute the rank of a name within a sex and year? (John was the nth most popular boys name in 2008...)

This task is easy if we have a single year & sex, but hard otherwise.

How would you do it for a single group?

```
one <- subset(bnames, sex == "boy" & year == 2008)
one$rank <- rank(-one$prop,
  ties.method = "first")

# or
one <- mutate(one,
  rank = rank(-prop, ties.method = "min"))
head(one)
```

What if we want to mutate every sex and year?

Workflow

1. Extract a single group
2. Figure out how to solve it for just that group
3. Use `ddply` to solve it for all groups

Workflow

1. Extract a single group
2. Figure out how to solve it for just that group
3. Use `ddply` to solve it for all groups

How would you use ddply to calculate all ranks?

```
bnames <- ddply(bnames, c("sex", "year"), mutate,
  rank = rank(-prop, ties.method = "min"))
```

`ddply + mutate =`
group-wise transformation

`ddply + summarise =`
per-group summaries

`ddply + subset =`
per-group subsets

Tools

You now have all the tools to solve 95% of data manipulation problems in R. It's just a matter of figuring out which tools to use, and how to combine them.

The following challenges will give you some practice.

Challenges

Warmups

Which names were most popular in 1999?

Work out the average yearly usage of each name.

List the 10 names with the highest average proportions.

```
# Which names were most popular in 1999?
subset(bnames, year == 1999 & rank < 10)
n1999 <- subset(bnames, year == 1999)
head(arrange(n1999, desc(prop)), 10)

# Average usage
overall <- ddply(bnames, "name", summarise,
  prop1 = mean(prop),
  prop2 = sum(prop) / 129)

# Top 10 names
head(arrange(overall, desc(prop)), 10)
```

Challenge 1

How has the total proportion of babies with names in the top 1000 changed over time?

How has the popularity of different initials changed over time?

```
sy <- ddply(bnames, c("year","sex"), summarise,
  prop = sum(prop),
  npop = sum(prop > 1/1000))

qplot(year, prop, data = sy, colour = sex,
  geom = "line")
qplot(year, npop, data = sy, colour = sex,
  geom = "line")
```

```
init <- ddply(bnames, c("year","first"), summarise,
  prop = sum(prop)/2)
```

```
qplot(year, prop, data = init, colour = first,
  geom = "line")
```

Challenge 2

For each name, find the year in which it was most popular, and the rank in that year. (Hint: you might find `which.max` useful).

Print all names that have been the most popular name at least once.

```
most_pop <- ddply(bnames, "name", summarise,
  year = year[which.max(prop)],
  rank = min(rank))
most_pop <- ddply(bnames, "name", subset,
  prop == max(prop))

subset(most_pop, rank == 1)

# Double challenge: Why is this last one wrong?
```

Challenge 3

What name has been in the top 10 most often?

(Hint: you'll have to do this in three steps. Think about what they are before starting)

```
top10 <- subset(bnames, rank <= 10)
counts <- count(top10, c("sex", "name"))
```

```
ddply(counts, "sex", subset, freq == max(freq))
head(arrange(counts, desc(freq)), 10)
```

Challenge 4

For each soundex, find the most common name in that group

```
names <- count(bnames2, c("soundex", "name"), "n")
```

```
ddply(names, "soundex", subset, freq == max(freq))
```