

polyr  
US baby names

Hadley Wickham

1. Introduction to the data
2. Transformations and summaries
3. Group-wise transformation and summary
4. Variable selection syntax
5. Challenge

# Baby names

Top 1000 male and female baby names in the US, from 1880 to 2008.

258,000 records ( $1000 * 2 * 129$ )

But only four variables: year, name, sex and percent.

```
> head(bnames, 15)
```

	year	name	percent	sex
1	1880	John	0.081541	boy
2	1880	William	0.080511	boy
3	1880	James	0.050057	boy
4	1880	Charles	0.045167	boy
5	1880	George	0.043292	boy
6	1880	Frank	0.027380	boy
7	1880	Joseph	0.022229	boy
8	1880	Thomas	0.021401	boy
9	1880	Henry	0.020641	boy
10	1880	Robert	0.020404	boy
11	1880	Edward	0.019965	boy
12	1880	Harry	0.018175	boy
13	1880	Walter	0.014822	boy
14	1880	Arthur	0.013504	boy
15	1880	Fred	0.013251	boy

```
> tail(bnames, 15)
```

	year	name	percent	sex
257986	2008	Neveah	0.000130	girl
257987	2008	Amaris	0.000129	girl
257988	2008	Hadassah	0.000129	girl
257989	2008	Dania	0.000129	girl
257990	2008	Hailie	0.000129	girl
257991	2008	Jamiya	0.000129	girl
257992	2008	Kathy	0.000129	girl
257993	2008	Laylah	0.000129	girl
257994	2008	Riya	0.000129	girl
257995	2008	Diya	0.000128	girl
257996	2008	Carleigh	0.000128	girl
257997	2008	Iyana	0.000128	girl
257998	2008	Kenley	0.000127	girl
257999	2008	Sloane	0.000127	girl
258000	2008	Elianna	0.000127	girl

# Brainstorm

What variables and summaries might you want to generate from this data? What questions would you like to be able to answer about the data?

With your partner, you have 2 minutes to come up with as many as possible.

# Some of my ideas

- First/last letter
- Length
- Number/percent of vowels
- Biblical names?
- Rank
- Ecdf (how many babies have a name in the top 2, 3, 5, 100 etc)

# Transform & summarise

```
transform(df, var1 = expr1, ...)
```

```
summarise(df, var1 = expr1, ...)
```

**Transform** modifies an existing data frame. **Summarise** creates a new data frame.

```
letter <- function(x, n = 1) {  
  if (n < 0) {  
    nc <- nchar(x)  
    n <- nc + n + 1  
  }  
  tolower(substr(x, n, n))  
}  
vowels <- function(x) {  
  nchar(gsub("[^aeiou]", "", x))  
}
```

```
bnames <- transform(bnames,  
  first = letter(name, 1),  
  last = letter(name, -1),  
  length = nchar(name),  
  vowels = vowels(name)  
)
```

```
summarise(bnames,  
  max_perc = max(percent),  
  min_perc = min(percent))
```

Many interesting transformations and summaries can be calculated for the whole dataset

# Group-wise

What about group-wise **transformations** or **summaries**? e.g. what if we want to compute the rank of a name within a sex and year?

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

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

```
# or
```

```
one <- transform(one,
  rank = rank(-percent, ties.method = "first"))
head(one)
```

What if we want to transform every sex and year?

```
# Split
```

```
pieces <- split(bnames,  
  list(bnames$sex, bnames$year))
```

```
# Apply
```

```
results <- vector("list", length(pieces))  
for(i in seq_along(pieces)) {  
  piece <- pieces[[i]]  
  piece <- transform(piece,  
    rank = rank(-percent, ties.method = "first"))  
  results[[i]] <- piece  
}
```

```
# Combine
```

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

```
# Or equivalently
```

```
bnames <- ddp1y(bnames, c("sex", "year"), transform,  
  rank = rank(-percent, ties.method = "first"))
```

Input data

Way to split  
up input

Function to apply to  
each piece

# Or equivalently

```
bnames <- ddply(bnames, c("sex", "year"), transform,  
rank = rank(-percent, ties.method = "first"))
```

2<sup>nd</sup> argument  
to transform()

# ddply

- `.data`: data frame to process
- `.variables`: combination of variables to split by
- `.fun`: function to call on each piece
- `...` : extra arguments passed to `.fun`

# Variable specification syntax

- Character: `c("sex", "year")`
- Numeric: `1:3`
- Formula: `~ sex + year`
- Special:
  - `.(sex, year)`
  - `.(first = letter(name, 1))`

# Match function with use

<code>scale(x)</code>	randomisation/permutation tests
<code>rank(x)</code>	scale to [0, 1] within each group
$x - \min(x) / \text{diff}(\text{range}(x))$	scale to mean 0, sd 1 within each group
$x / x[1]$	compute per-group rankings
<code>sample(x)</code>	index a time series

# Summaries

In a similar way, we can use `ddply()` for group-wise summaries.

There are many base R functions for special cases. Where available, these are often much faster; but you have to know they exist, and have to remember how to use them.

```
ddply(bnames, c("name"), summarise,  
      tot = sum(percent))  
ddply(bnames, c("length"), summarise,  
      tot = sum(percent))  
ddply(bnames, c("year", "sex"), summarise,  
      tot = sum(percent))  
  
f1 <- ddp1y(bnames, c("year", "sex", "first"),  
           summarise, tot = sum(percent))  
library(ggplot2)  
qplot(year, tot, data = f1, geom = "line",  
       colour = sex, facets = ~ first)
```

# Challenge

Create a plot that shows (by year) the proportion of US children who have a name in the top 100.

Extra challenge: break it down by sex.

What does this suggest about baby naming trends in the US?

