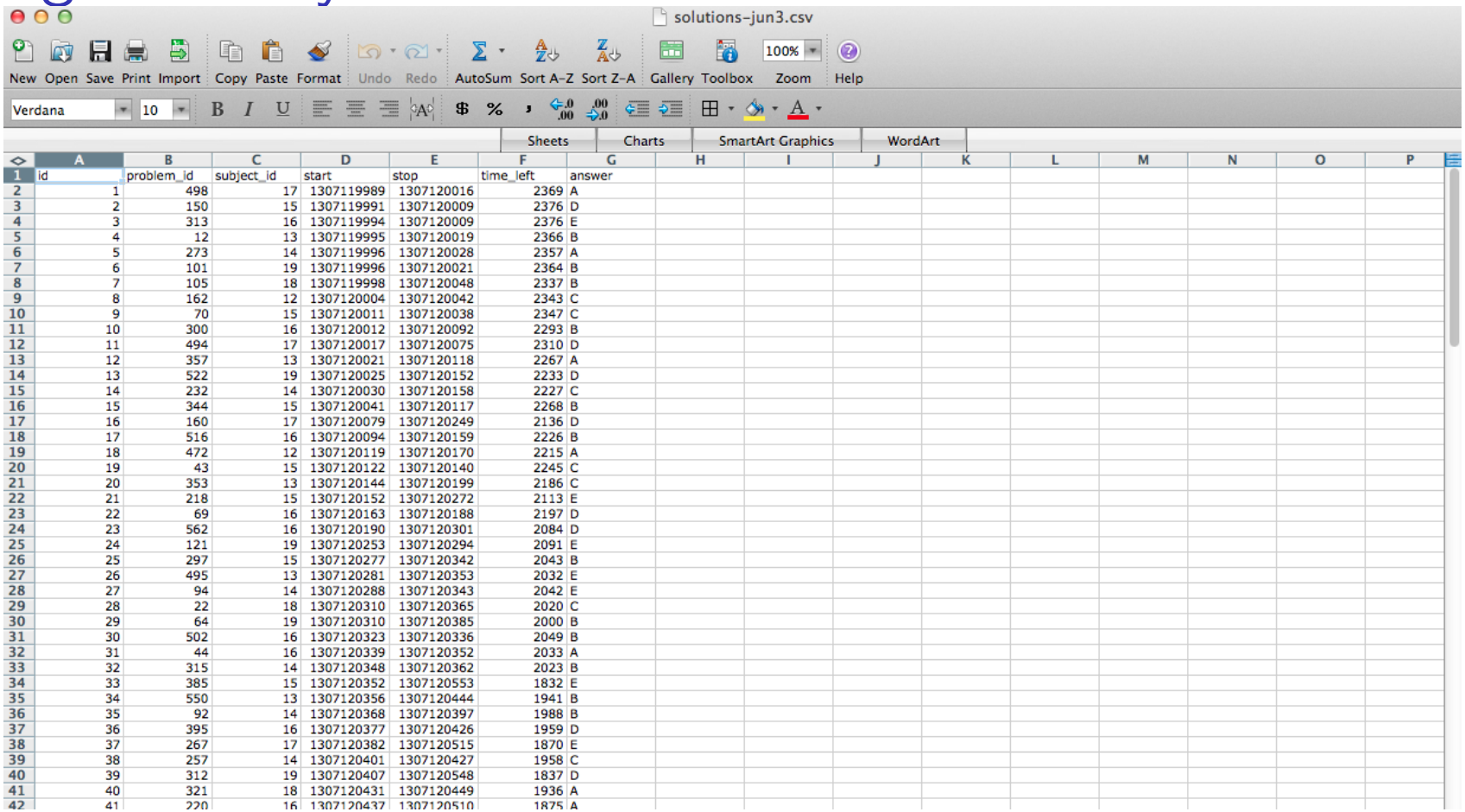
Reshaping data

Jeffrey Leek

May 18, 2016

The goal is tidy data



	A	B	C	D	E	F	G
1	id	problem_id	subject_id	start	stop	time_left	answer
2	1	498	17	1307119989	1307120016	2369	A
3	2	150	15	1307119991	1307120009	2376	D
4	3	313	16	1307119994	1307120009	2376	E
5	4	12	13	1307119995	1307120019	2366	B
6	5	273	14	1307119996	1307120028	2357	A
7	6	101	19	1307119996	1307120021	2364	B
8	7	105	18	1307119998	1307120048	2337	B
9	8	162	12	1307120004	1307120042	2343	C
10	9	70	15	1307120011	1307120038	2347	C
11	10	300	16	1307120012	1307120092	2293	B
12	11	494	17	1307120017	1307120075	2310	D
13	12	357	13	1307120021	1307120118	2267	A
14	13	522	19	1307120025	1307120152	2233	D
15	14	232	14	1307120030	1307120158	2227	C
16	15	344	15	1307120041	1307120117	2268	B
17	16	160	17	1307120079	1307120249	2136	D
18	17	516	16	1307120094	1307120159	2226	B
19	18	472	12	1307120119	1307120170	2215	A
20	19	43	15	1307120122	1307120140	2245	C
21	20	353	13	1307120144	1307120199	2186	C
22	21	218	15	1307120152	1307120272	2113	E
23	22	69	16	1307120163	1307120188	2197	D
24	23	562	16	1307120190	1307120301	2084	D
25	24	121	19	1307120253	1307120294	2091	E
26	25	297	15	1307120277	1307120342	2043	B
27	26	495	13	1307120281	1307120353	2032	E
28	27	94	14	1307120288	1307120343	2042	E
29	28	22	18	1307120310	1307120365	2020	C
30	29	64	19	1307120310	1307120385	2000	B
31	30	502	16	1307120323	1307120336	2049	B
32	31	44	16	1307120339	1307120352	2033	A
33	32	315	14	1307120348	1307120362	2023	B
34	33	385	15	1307120352	1307120553	1832	E
35	34	550	13	1307120356	1307120444	1941	B
36	35	92	14	1307120368	1307120397	1988	B
37	36	395	16	1307120377	1307120426	1959	D
38	37	267	17	1307120382	1307120515	1870	E
39	38	257	14	1307120401	1307120427	1958	C
40	39	312	19	1307120407	1307120548	1837	D
41	40	321	18	1307120431	1307120449	1936	A
42	41	220	16	1307120437	1307120510	1875	A

1. Each variable forms a column
2. Each observation forms a row
3. Each table/file stores data about one kind of observation (e.g. people/hospitals).

Start with reshaping

```
library(reshape2)
library(plyr)
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs
## Mazda RX4        21.0   6  160 110 3.90 2.620 16.46  0
## Mazda RX4 Wag    21.0   6  160 110 3.90 2.875 17.02  0
## Datsun 710       22.8   4  108  93 3.85 2.320 18.61  1
## Hornet 4 Drive   21.4   6  258 110 3.08 3.215 19.44  1
## Hornet Sportabout 18.7  8  360 175 3.15 3.440 17.02  0
## Valiant          18.1   6  225 105 2.76 3.460 20.22  1
```

Melting data frames

```
mtcars$carname <- rownames(mtcars)
carMelt <- melt(mtcars,id=c("carname","gear","cyl"),measure
head(carMelt,n=3)
```

```
##           carname gear cyl variable value
## 1       Mazda RX4    4   6      mpg  21.0
## 2 Mazda RX4 Wag    4   6      mpg  21.0
## 3    Datsun 710    4   4      mpg  22.8
```

```
tail(carMelt,n=3)
```

```
##            carname gear cyl variable value
## 62  Ferrari Dino    5   6       hp   175
## 63 Maserati Bora    5   8       hp   335
## 64    Volvo 142E    4   4       hp   109
```

http://www.statmethods.net/management/reshape.html

Casting data frames

```
cylData <- dcast(carMelt, cyl ~ variable)
```

```
## Aggregation function missing: defaulting to length
```

```
cylData
```

```
##   cyl mpg hp
## 1   4  11 11
## 2   6   7  7
## 3   8  14 14
```

```
cylData <- dcast(carMelt, cyl ~ variable,mean)
cylData
```

```
##   cyl      mpg       hp
## 1   4 26.66364 82.63636
## 2   6 19.74286 122.28571
## 3   8 15.10000 209.21429
```

Averaging values

```
head(InsectSprays)
```

```
##    count spray
## 1     10     A
## 2      7     A
## 3     20     A
## 4     14     A
## 5     14     A
## 6     12     A
```

```
tapply(InsectSprays$count,InsectSprays$spray,sum)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

http://www.r-bloggers.com/
a-quick-primer-on-split-apply-combine-problems/

Another way - split

```
spIns =  split(InsectSprays$count,InsectSprays$spray)
spIns
```

```
## $A
##  [1] 10  7 20 14 14 12 10 23 17 20 14 13
##
## $B
##  [1] 11 17 21 11 16 14 17 17 19 21  7 13
##
## $C
##  [1] 0 1 7 2 3 1 2 1 3 0 1 4
##
## $D
##  [1]  3  5 12  6  4  3  5  5  5  5  2  4
##
## $E
##  [1] 3 5 3 5 3 6 1 1 3 2 6 4
##
```

Another way - apply

```
sprCount = lapply(spIns,sum)
sprCount
```

```
## $A
## [1] 174
##
## $B
## [1] 184
##
## $C
## [1] 25
##
## $D
## [1] 59
##
## $E
## [1] 42
##
```

Another way - combine

```
unlist(sprCount)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

```
sapply(spIns,sum)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

Another way - plyr package

```
ddply(InsectSprays,.(spray),summarize,sum=sum(count))
```

```
##   spray sum
## 1     A 174
## 2     B 184
## 3     C  25
## 4     D  59
## 5     E  42
## 6     F 200
```

Creating a new variable

```
spraySums <- ddply(InsectSprays,.(spray),summarize,sum=ave
dim(spraySums)
```

```
## [1] 72  2
```

```
head(spraySums)
```

```
##    spray sum
## 1      A 174
## 2      A 174
## 3      A 174
## 4      A 174
## 5      A 174
## 6      A 174
```

More information

- A tutorial from the developer of plyr - `http://plyr.had.co.nz/09-user/`
- A nice reshape tutorial `http://www.slideshare.net/jeffreybreen/reshaping-data-in-r`
- A good plyr primer - `http://www.r-bloggers.com/a-quick-primer-on-split-apply-combine-problems/`
- See also the functions
- acast - for casting as multi-dimensional arrays
- arrange - for faster reordering without using order() commands
- mutate - adding new variables