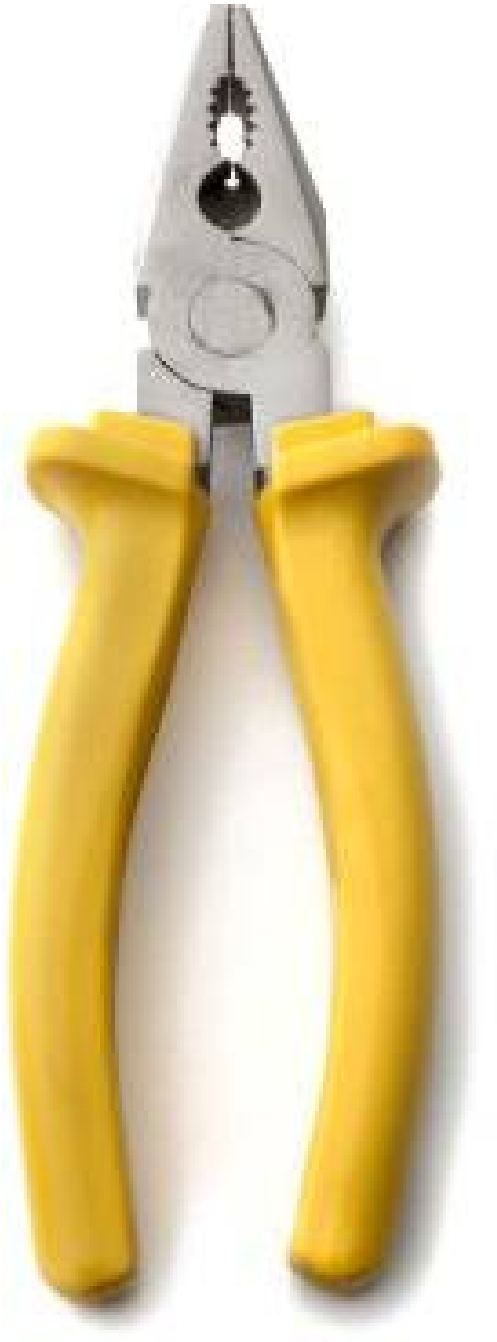
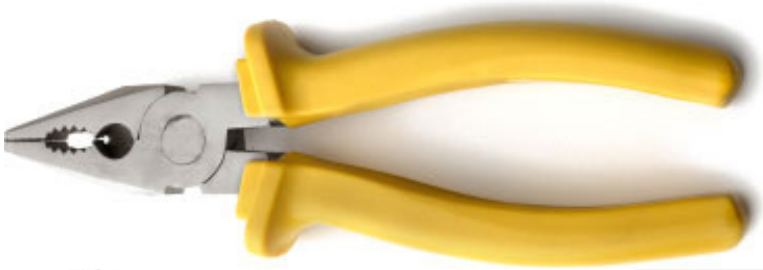


KICKING ASS WITH plyr

The Taking of Names is Optional

JD Long (@CMastication) on the web at
http://CerebralMastication.com



Who Is This Guy?

stackoverflow Questions Tags Users Badges Unanswered

Tagged Questions stats newest fe

Top r Answerers

Last 30 Days

104	33	Shane	2,799 ●2 ●12
51	14	Dirk Eddelbuettel	6,919 ●3 ●16
48	12	Rob Hyndman	1,612 ●2 ●11
30	9	Harlan	511 ●7
28	8	rcs	992 ●1 ●10
25	4	Ian Fellows	490 ●8
21	5	Jonathan Chang	689 ●1 ●6

All time

356	101	Dirk Eddelbuettel	6,919 ●3 ●16
209	74	Shane	2,799 ●2 ●12
127	38	hadley	1,612 ●2 ●8
91	26	chris_dubois	1,484 ●12
83	27	ars	5,762 ●3 ●9
81	27	Rob Hyndman	1,612 ●2 ●11
77	20	Eduardo Leoni	1,151 ●1 ●8

Dirk

Hadley

Top r Askers

Last 30 Days

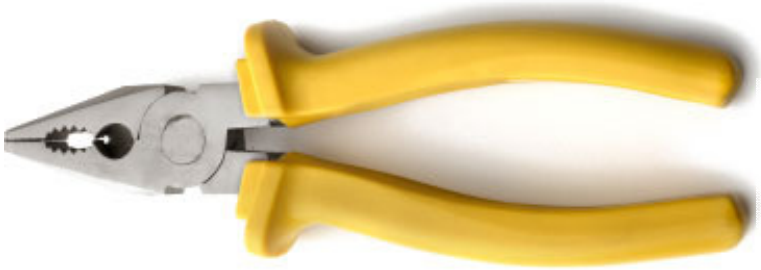
10	2	Shane	2,799 ●2 ●12
9	3	knot	117 ●5
7	3	andrewj	161 ●6
7	3	Vince	914 ●1 ●12

All time

51	24	JD Long	1,530 ●1 ●11
47	20	chris_dubois	1,484 ●12
42	13	Dan Goldstein	619 ●1 ●11
32	8	gappy	799 ●10

Me!

http://stackoverflow.com/questions/tagged/r



What Does plyr Do?

It Does:

Split

Apply

Combine

It's Like:

SQL Group By:

```
SELECT
        dim1, dim2,
        sum(fact)
FROM
        MyData
GROUP BY
        dim1, dim2
```

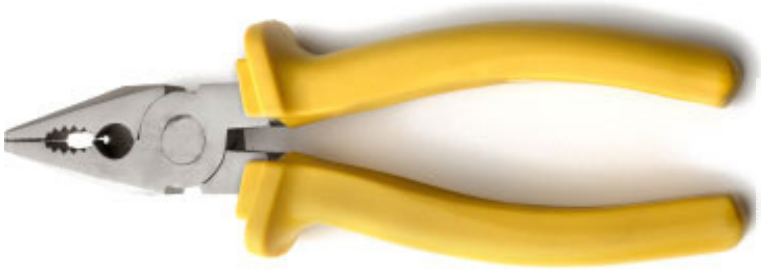
or

Excel PivotTable:

Sum of fact	Column Labels	
Row Labels	foo	that
that	8	6
this	5	8

This is nothing new to R

plyr is an easy, consistent wrapper without loops



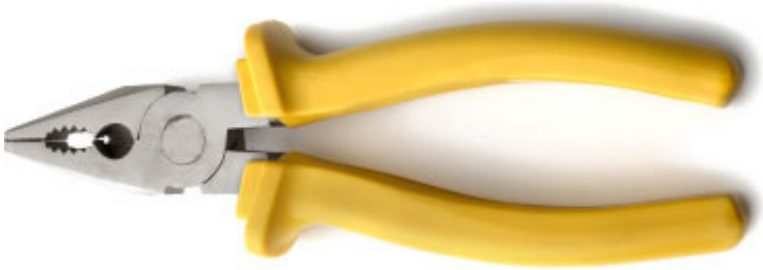
Basic Syntax Example

```
library(plyr)

#make some example data
dd<- data.frame(matrix(rnorm(216),72,3), c(rep("A",24),rep("B",24),
rep("C",24)),c(rep("J",36), rep("K",36)))
colnames(dd) <- c("v1", "v2", "v3", "dim1", "dim2")

ddply(dd, c("dim1","dim2"), function(df) mean(df$v1))
  #or#
ddply(dd, c("dim1","dim2"), summarise, V1=mean(v1))
```

Output:

dim1	dim2	V1
A	J	-0.05777168
B	J	-0.16168041
B	K	0.55458059
C	K	0.26616421

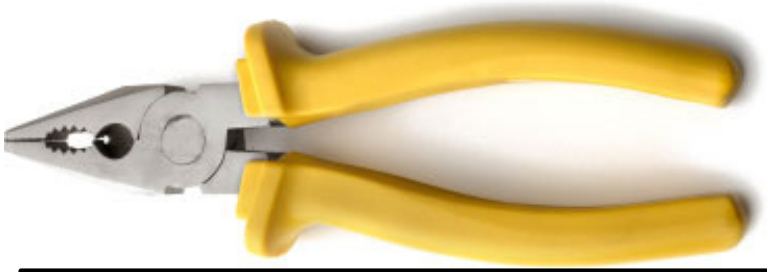
This example and more: http://www.cerebralmastication.com/?p=339



What does dd mean?

Object In		Object Out		
a (array) d (data frame) l (list)	+	a (array) d (data frame) l (list) _ (nothing)	+	**ply**

ddply – data frame in, data frame out
ldply – list in, data frame out
alply – array in, list out
d_ply – data frame in, no object out (useful for plotting or file output)
… – http://had.co.nz/plyr/plyr-intro-090510.pdf (plyr manual)



Another Example

Picking up where the other example left off:

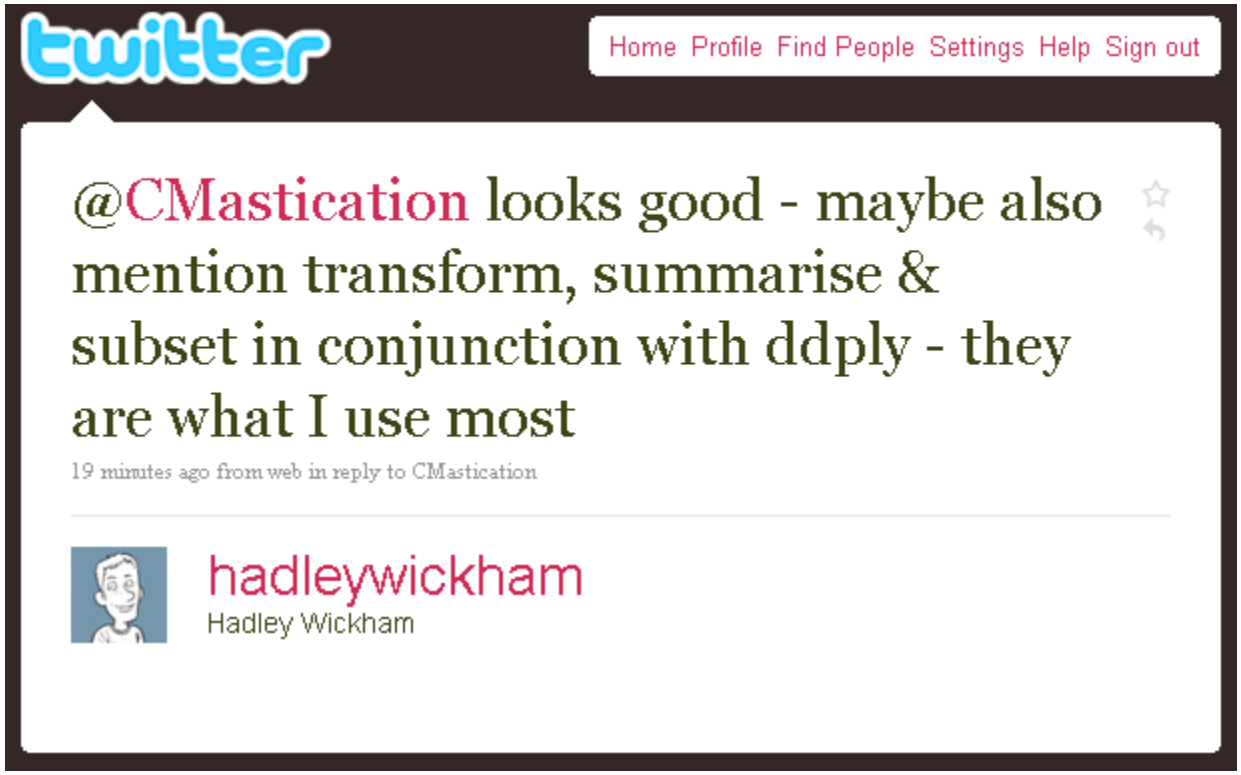
```
library(Hmisc) #the weighted ECDF function is in here
```

Create a weighted empirical cumulative distribution function for each combination of dim1 and dim2

```
dlply(dd, c("dim1", "dim2" ), function(df) wtd.Ecdf(df$v1, df$v2 ))
```

Output:

```
$A.J
$A.J$x
 [1] -2.3231636 -2.3231636 -2.1657044 -1.6740506 -1.5489670 -1.2253579 -0.7212321
 [8] -0.6582277 -0.5071541 -0.4944474 -0.4637418 -0.3948859  0.0967614  0.2608770
[15]  0.3031230  0.4634211  0.6252545  0.6636364  0.7422236  0.8453430  0.8935186
[22]  1.0230601  1.1296187  1.8608002  1.8827746

$A.J$ecdf
 [1] 0.00000000 0.04210145 0.08396510 0.12781269 0.16723571 0.20641107 0.24805195
 [8] 0.28895378 0.32858999 0.37249865 0.41181383 0.45459172 0.49844146 0.54268038
[15] 0.58489520 0.62717256 0.66780344 0.70886233 0.74951405 0.78984539 0.83499854
[22] 0.87609499 0.91933162 0.95915316 1.00000000

...
```



plyr web site: http://had.co.nz/plyr/
examples and drills: http://had.co.nz/stat405/resources/drills/plyr.html

I stole the big fat pliers graphic from the plyr web site