Using data.table

Jeffrey Leek

May 18, 2016

data.table

- Inherets from data.frame
- All functions that accept data.frame work on data.table
- Written in C so it is much faster
- Much, much faster at subsetting, group, and updating

Create data tables just like data frames

```
library(data.table)
DF = data.frame(x=rnorm(9),y=rep(c("a","b","c"),each=3),z=r
head(DF,3)
```

```
##             x y          z
## 1  1.16073446 a 0.1540230
## 2 -0.08584376 a 1.5045276
## 3 -0.79074538 a 0.3529631
```

```
DT = data.table(x=rnorm(9),y=rep(c("a","b","c"),each=3),z=r
head(DT,3)
```

```
##             x y          z
## 1: -0.05698985 a 0.7454435
## 2:  0.15306681 a 0.3452477
## 3: -1.57382282 a 0.8278485
```

See all the data tables in memory

```
tables()
```

```
##      NAME NROW NCOL MB COLS  KEY
## [1,] DT      9    3  1 x,y,z
## Total: 1MB
```

Subsetting rows

```
DT[2,]
```

```
##             x y         z
## 1: 0.1530668 a 0.3452477
```

```
DT[DT$y=="a",]
```

```
##              x y         z
## 1: -0.05698985 a 0.7454435
## 2:  0.15306681 a 0.3452477
## 3: -1.57382282 a 0.8278485
```

Subsetting rows

```
DT[c(2,3)]
```

```
##             x y         z
## 1:  0.1530668 a 0.3452477
## 2: -1.5738228 a 0.8278485
```

Subsetting columns!?

```
DT[,c(2,3)]
```

```
## [1] 2 3
```

Column subsetting in data.table

- The subsetting function is modified for data.table
- The argument you pass after the comma is called an "expression"
- In R an expression is a collection of statements enclosed in curley brackets

```
{
  x = 1
  y = 2
}
k = {print(10); 5}
```

```
## [1] 10
```

```
print(k)
```

```
## [1] 5
```

Calculating values for variables with expressions

```
DT[,list(mean(x),sum(z))]
```

```
##                V1        V2
## 1: -0.09469935 6.144561
```

```
DT[,table(y)]
```

```
## y
## a b c
## 3 3 3
```

Adding new columns

```
DT[,w:=z^2]
```

Adding new columns

```
DT2 <- DT
DT[, y:= 2]
```

```
## Warning in `[.data.table`(DT, , `:=`(y, 2)): Coerced 'do
## 'character' to match the column's type; may have truncat
## Either change the target column to 'double' first (by cr
## 'double' vector length 9 (nrows of entire table) and ass
## 'replace' column), or coerce RHS to 'character' (e.g. 1L
## integer]_, as.*, etc) to make your intent clear and for
## column type correctly up front when you create the table
## please.
```

Careful

```
head(DT,n=3)
```

```
##                x y         z         w
## 1: -0.05698985 2 0.7454435 0.5556861
## 2:  0.15306681 2 0.3452477 0.1191960
## 3: -1.57382282 2 0.8278485 0.6853332
```

```
head(DT2,n=3)
```

```
##                x y         z         w
## 1: -0.05698985 2 0.7454435 0.5556861
## 2:  0.15306681 2 0.3452477 0.1191960
## 3: -1.57382282 2 0.8278485 0.6853332
```

Multiple operations

```
DT[,m:= {tmp <- (x+z); log2(tmp+5)}]
```

plyr like operations

```
DT[,a:=x>0]
```

plyr like operations

```
DT[,b:= mean(x+w),by=a]
```

Special variables

`.N` An integer, length 1, containing the number of elements of a factor level

```
set.seed(123);
DT <- data.table(x=sample(letters[1:3], 1E5, TRUE))
DT[, .N, by=x]
```

```
##    x     N
## 1: a 33387
## 2: c 33201
## 3: b 33412
```

Keys

```
DT <- data.table(x=rep(c("a","b","c"),each=100), y=rnorm(30
setkey(DT, x)
DT['a']
```

```
##       x           y
##    1: a  0.25958973
##    2: a  0.91751072
##    3: a -0.72231834
##    4: a -0.80828402
##    5: a -0.14135202
##    6: a  2.25701345
##    7: a -2.37955015
##    8: a -0.45425393
##    9: a -0.06007418
##   10: a  0.86090061
##   11: a -1.78466393
##   12: a -0.13074225
##   13: a -0.36983749
```

Joins

```
DT1 <- data.table(x=c('a', 'a', 'b', 'dt1'), y=1:4)
DT2 <- data.table(x=c('a', 'b', 'dt2'), z=5:7)
setkey(DT1, x); setkey(DT2, x)
merge(DT1, DT2)
```

```
##    x y z
## 1: a 1 5
## 2: a 2 5
## 3: b 3 6
```

Fast reading

```
big_df <- data.frame(x=rnorm(1E6), y=rnorm(1E6))
file <- tempfile()
write.table(big_df, file=file, row.names=FALSE, col.names=T
system.time(fread(file))
```

```
##    user  system elapsed
##   0.424   0.026   0.468
```

```
system.time(read.table(file, header=TRUE, sep="\t"))
```

```
##    user  system elapsed
##  10.417   0.168  10.837
```

Summary and further reading

- The latest development version contains new functions like `melt` and `dcast` for data.tables
- `https://r-forge.r-project.org/scm/viewvc.php/pkg/NEWS?view=markup&root=datatable`
- Here is a list of differences between data.table and data.frame
- `http://stackoverflow.com/questions/13618488/what-you-can-do-with-data-frame-that-you-cant-in-data-`
- Notes based on Raphael Gottardo's notes `https://github.com/raphg/Biostat-578/blob/master/Advanced_data_manipulation.Rpres`, who got them from Kevin Ushey.