data.table

16 September 2014

EARL, London

Matt Dowle

v2

This talk contains quotes from real people.

Joe: ”text” = text appears verbatim in public

Joe: text = I am paraphrasing

1996

I graduate in Maths and Computing

Join Lehman Brothers, London

VB/Excel and Sybase SQL

1999

Move to Salomon Brothers, London

On Day 1 I meet Patrick Burns; author of S Poetry and now R Inferno

Pat: We use S-PLUS here.

Matt: What's S-PLUS?

Pat shows me S-PLUS

```
> DF <- data.frame(
          A = letters[1:3],
          B = c(1,3,5)       )
> DF
  A B
1 a 1
2 b 3
3 c 5
```

Already easier than SQL

Pat: Columns. Same length. Different types.

Matt: Like a database table?

Pat: Yes

Matt: Great. I get it. You didn't have to do CREATE TABLE first and then INSERT data?

Pat: Correct. It's one step.

Matt: Show me more!

Rows are stored in order

Pat:

```
> DF[2:3,]
  A B
2 b 3
3 c 5
```

Matt: WOW! I don't need to create a column containing row numbers like I do in SQL?

Pat: Nope. The row order is how it's stored in memory. That's why it's good for time series.

My first thought

```
> DF[2:3,]
  A B
2 b 3
3 c 5
```

```
> DF[2:3, sum(B)]
[1] 8
```

No

Pat: Ah, no.

Matt: Why not?

Pat: It's **`sum(DF[2:3,"B"])`**

Matt: Ok, but why not what I tried?

Pat: It doesn't work like that.

Matt: Why not?

Pat: Because it doesn't.

Matt: What does it do then?

Pat: Nothing, don't do it.

Matt: I tried it anyway. It's an error.

```
object 'B' not found
```

Pat: Yeah I told you not to do that.

Matt: Can we ask S-PLUS to change it?

Pat: Good luck!

Matt: Ok ok. I'll move on.

3 years pass, 2002

One day S-PLUS crashes

It's not my code, but a corruption in S-PLUS

Support: Are you sure it's not *your* code.

Matt: Yes. See, here's how you reproduce it.

Support: Yes, you're right. We'll fix it, thanks!

Matt: Great, when?

When

Support: Immediately. For the next release.

Matt: Great, when's that?

Support: 6 months

Matt: Can you do a patch quicker?

Support: No because it's just you with the problem.

Matt: But I'm at Salomon/Citigroup, the biggest financial corporation in the world!

Support: True but it's still just you, Matt.

When (continued)

Matt: I understand. Can you send me the code and I'll fix it? I don't mind - I'll do it for free. I just want to fix it to get my job done.

Support: Sorry, can't do that. Lawyer says no.

Matt: Pat, any ideas?

Pat: Have you tried R?

Matt: What's R?

R in 2002

I took the code I had in S-PLUS and ran it in R.

Not only didn't it crash, but it took 1 minute instead of 1 hour.

R had improved the speed of `for` loops (*) and was in-memory rather than on-disk.

(*) The code generated random portfolios and couldn't be vectorized, due to its nature.

Even better

If R does error or crash, I can fix it. We have the source code! Or we can hire someone to fix it for us.

I can make progress and not wait 6 months for a fix. I have options.

And it has **packages**.

I start to use R.

My first thought, again

Matt: Pat, remember how I first thought **`[.data.frame`** should work?

```
DF[2:3, sum(B)]
```

Pat: Ha ha. Good luck!

2004, day 1

DF[2:3, sum(B)] is born.

Only possible because R (uniquely) has lazy evaluation.

2004, day 2

I do the same for `i`

DF[type=="books", sum(sales)]

2004, day 3

I realise I need group by :

DF[type=="books", sum(sales), by=country]

2004, day 4

I realise **chaining** comes for free:

DF[, sum(sales), by=country][order(-V1),]

Dec 2005 : Problem

Matt to r-help : Why is **DF** 10 times bigger and 10 times slower than **M**?

n = 1,000,000

M = matrix(integer(n), nrow=n, ncol=2)

0.22 secs to create 7MB

DF = data.frame(a=integer(n), b=integer(n))

2.81 secs to create 76MB

Row names

Peter Dalgaard on r-help :

”

Row names!!

r <- as.character(1:1e6)

object.size(r)/1024^2

[1] 68 MB [90% of 76MB]

”

What to do?

Matlab?

No data.frame. I already have Matlab.

Python?

No data.frame. No CRAN.

See if S-PLUS is faster now?

Firm already has Matlab.

Fix R?

Why not?

Dec 2005

Matt on r-devel : Following up on that r-help thread, here's some code attached that doesn't create rownames, can data.frame be changed?

Prof Brian Ripley : ”Data frames have unique row names *by definition* (White Book p.57).

Note that R is extensible, so any package writer has (for 14 years since the White Book) been entitled to assume that. A minimum test suite is to run R CMD check on all CRAN packages, and to read all the relevant documentation. That would reveal a large number of uses of row names and of their uniqueness.”

12 Apr 2006

data.table 1.0 released to CRAN

” This package does very little … Just like a data.frame but without rownames, up to 10 times faster, up to 10 times less memory … allows subset() and with() like expressions inside DT[...].

The White Book defines rownames, so data.frame can't be changed => new class ”

cc'd r-devel

14 Apr 2006

Brian Ripley to r-devel: How about delaying creation of row names until they are really needed?

R-devel feedback: Great!

Brian Ripley: Done. Please test.

R 2.4.0 New Feature :

” The internal storage of row.names = 1:n just records 'n', for efficiency with very long vectors. ”

Job done!

I then removed data.table from CRAN. It was there about 2 weeks.

It had served its purpose. I'm delighted.

But … I continue to use data.table myself for the nice syntax :

DT[type==”books”, sum(sales), by=country]

Aug 2008

I release data.table 1.1 to CRAN :

DT[where, select, group by][...][...]

2009 : Tom Short

Tom: I like data.table! But setkey is slow and I have issues with dates. Any ideas, Matt?

Matt: Sorry, not really. I can have a think. You're welcome to join the project?

Tom: Ok thanks, I will. What about using **base::sort.list(...,method="radix")**?

Matt: What's **base::sort.list(...,method="radix")** ?

NB: base::sort.list() returns an order, like order() not sort().

Tom uses base::method=”radix”

nrow(DT) = 10 million	v1.2	v1.3
setkey(DT, a, b)	**37s**	**5s**

Column by column in reverse :

1. o = 1:nrow
2. order o by column b
3. order o by column a

Hard to beat, even today. *Provided* both columns are integer/factor, both with range < 100,000

method=”radix” is really a counting sort and we like it a lot

Jun 2011

Steve Lianoglou joins the project.

- data.table now plays nicely with S4 classes.
- New testthat framework
- Many bug fixes and improvements

Sub-assign by reference

In SQL you just change the SELECT keyword to be UPDATE instead.

I started to play with overloading `<-` , `=` or `<<-` in j but it wasn't ideal

Matt: Any ideas, Simon Urbanek?

Simon Urbanek: How about `:=` for that?

Matt: What's `:=` ?

Aug 2011

I define `:=` in `j` to do assignment by reference, combined with subset

DT[where, select | update]

From v1.6.3 NEWS :

```
for (i in 1:1000) DF[i,1] <- i       # 591s
for (i in 1:1000) DT[i,V1:=i]        #     1s
```

User reaction

”data.table is awesome! That took about 3 seconds [was 30 mins] for the whole thing!!!”

Davy Kavanagh, 15 Jun 2012

2013, fast and friendly file reading

e.g. 50MB .csv, 1 million rows x 6 columns

```
read.csv("test.csv")              # 30-60s
read.csv("test.csv", colClasses=,
         nrows=, etc...)          #    10s
fread("test.csv")                 #     3s
```

e.g. 20GB .csv, 200 million rows x 16 columns

```
read.csv(”big.csv”, ...)          #  hours
fread("big.csv")                  #     8m
```

Aug 2013

More users start committing :

- Arun Srinivasan
  - Bug fixes and enhancements
- Eduard Antonyan
  - Adds fread("grep blah file.txt")
  - rbindlist retaining/combining factors
  - Bug fixes and enhancements
- Ricardo Saporta
  - Bug fixes and enhancements

Feb 2014

Arun has learnt C and data.table's C code

In C, he adds fast dcast and melt

Then Arun also adds true radix sorting using:

Terdiman, 2000:
http://codercorner.com/RadixSortRevisited.htm

Herf, 2001:
http://stereopsis.com/radix.html

I tweak it from LSD backwards to MSD forwards

This leads to GForce

Radix sort for numeric & character

20 million rows x 4 columns, 539MB

a & b (numeric), c (integer), d (character)

	v1.8.10	v1.9.2
setkey(DT, a)	54.9s	5.3s
setkey(DT, c)	48.0s	3.9s
setkey(DT, a, b)	102.3s	6.9s
”Cold” grouping (no setkey first) :		
DT[, mean(b), by=c]	47.0s	3.4s

https://gist.github.com/arunsrinivasan/451056660118628befff

New feature: melt

i.e. reshape2 for data.table

20 million rows x 6 columns (a:f) 768MB

melt(DF, id="d", measure=1:2) 4.1s (*)

melt(DT, id="d", measure=1:2) 1.7s

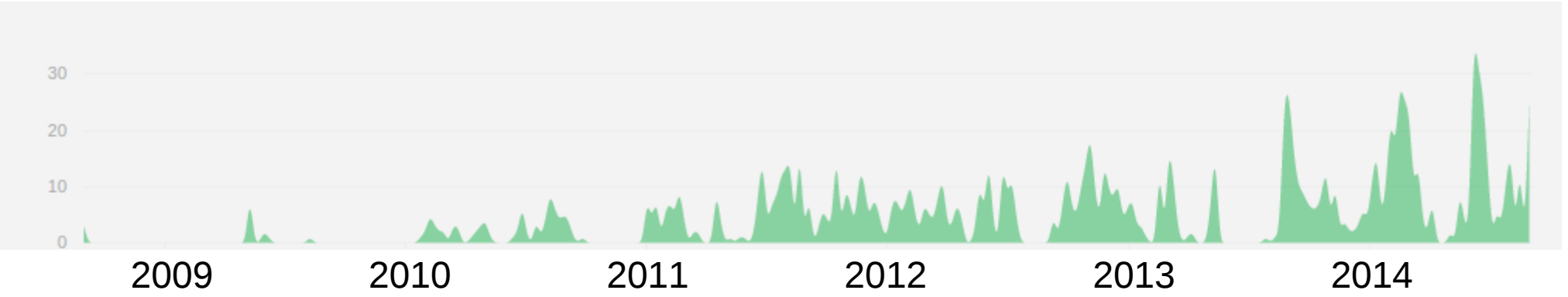
(*) including Kevin Ushey's C code in reshape2, was 190s

melt(DF, ..., na.rm=TRUE) 39.5s

melt(DT, ..., na.rm=TRUE) 2.7s

https://gist.github.com/arunsrinivasan/45105660118628befff

Commit history to date



Number of commits	
Matt	958
Arun	300
Tom	38
Steve	37
Ed	32
Rick	7

data.table answerers

Last 30 Days

19	10		eddi * **18.1k** ●2 ●17 ●45
15	7		David Arenburg **9,748** ●2 ●6 ●27
14	4		Roland **34.6k** ●3 ●24 ●54
11	5		Arun * **39.7k** ●6 ●42 ●92
9	4		BrodieG **13.8k** ●6 ●24
8	2		Ananda Mahto **67k** ●8 ●54 ●115
6	3		akrun **9,169** ●1 ●5 ●13

All Time

1.4k	215		Matt Dowle * **21k** ●4 ●48 ●93
922	218		Arun * **39.7k** ●6 ●42 ●92
766	159		mnel **48.2k** ●6 ●77 ●107
572	73		Josh O'Brien **66.2k** ●3 ●90 ●169
527	196		eddi * **18.1k** ●2 ●17 ●45
267	72		Ricardo Saporta * **21.8k** ●2 ●27 ●57
263	84		BondedDust **106k** ●3 ●66 ●148

Number of answers provided

Number of +1 votes for those answers

* code contributors

data.table support

16	Last 7 Days	43.8% unanswered
68	Last 30 Days	26.5% unanswered
1,726	All Time	9.2% unanswered

As of 10 Sep 2014

Testing

data.table has :

3,700 lines of R code

7,300 lines of C code

+ 3,400 tests

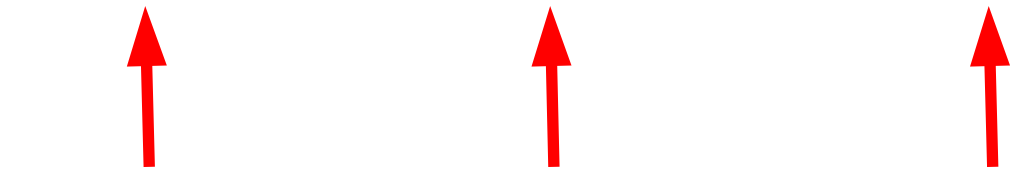
4,900 lines of test code

Run by CRAN every day

Includes tests with other packages
e.g. ggplot2, reshape2

Example tests

```
DT = data.table(a=1:6, b=7:12)
test(1348, DT[.N], DT[6])
test(1349, DT[.N-1:3], DT[5:3])
test(1350, DT[.N+1], DT[NA])
```



Whichever test framework you use, needs to be easy to read and easy to add new tests

Plus tests in dependent packages

- 52 CRAN packages

ALFQ	Causata	DataCombine	dplyr	ecoengine
edmr	eeptools	FAOSTAT	freqweights	gems
greport	IAT	installr	Kmisc	Lahman
lar	lllcrc	LogisticDx	optiRum	psidR
RAPIDR	rbison	Rbitcoin	rfisheries	rgauges
rgbif	rnoaa	rplos	SciencesPo	sdcMicro
sdcTable	SGP	simPH	spocc	survMisc
sweSCB	taxize	treebase	treemap	slackr
benford.analysis		randomNames	RecordLinkage	vardpoor
ProjectTemplate		CAGExploreR	splitstackshape	edgeRun

- 14 Bioconductor packages

CAGEr	COMPASS	flowWorkspace	GGtools
GOTHiC	MIMOSA	openCyto	phyloseq
QUALIFIER	R3Cseq	rBiopaxParser	rfPred
rTANDEM	RTN		

roll = ”nearest”

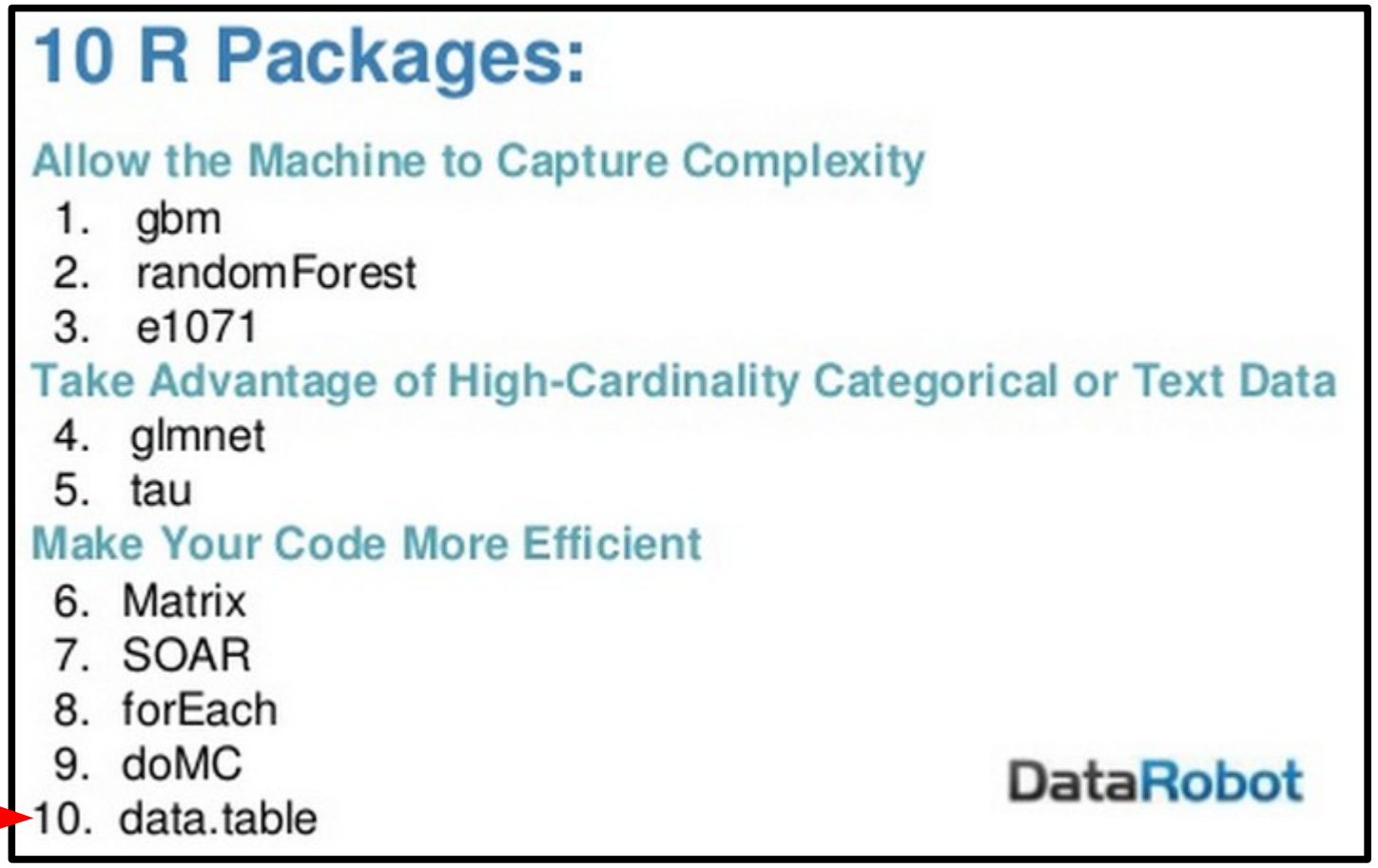
x	y	value
A	2	1.1
A	9	1.2
A	11	1.3
B	3	1.4

```
setkey(DT, x, y)
DT[.(”A”,7), roll=”nearest”]
```

+ forwards, backwards, limited and ends

”10 R packages to win Kaggle competitions”, useR! 2014

By Xavier Conort of DataRobot.com



data.table →

http://www.slideshare.net/DataRobot/final-10-r-xc-36610234

Why R?

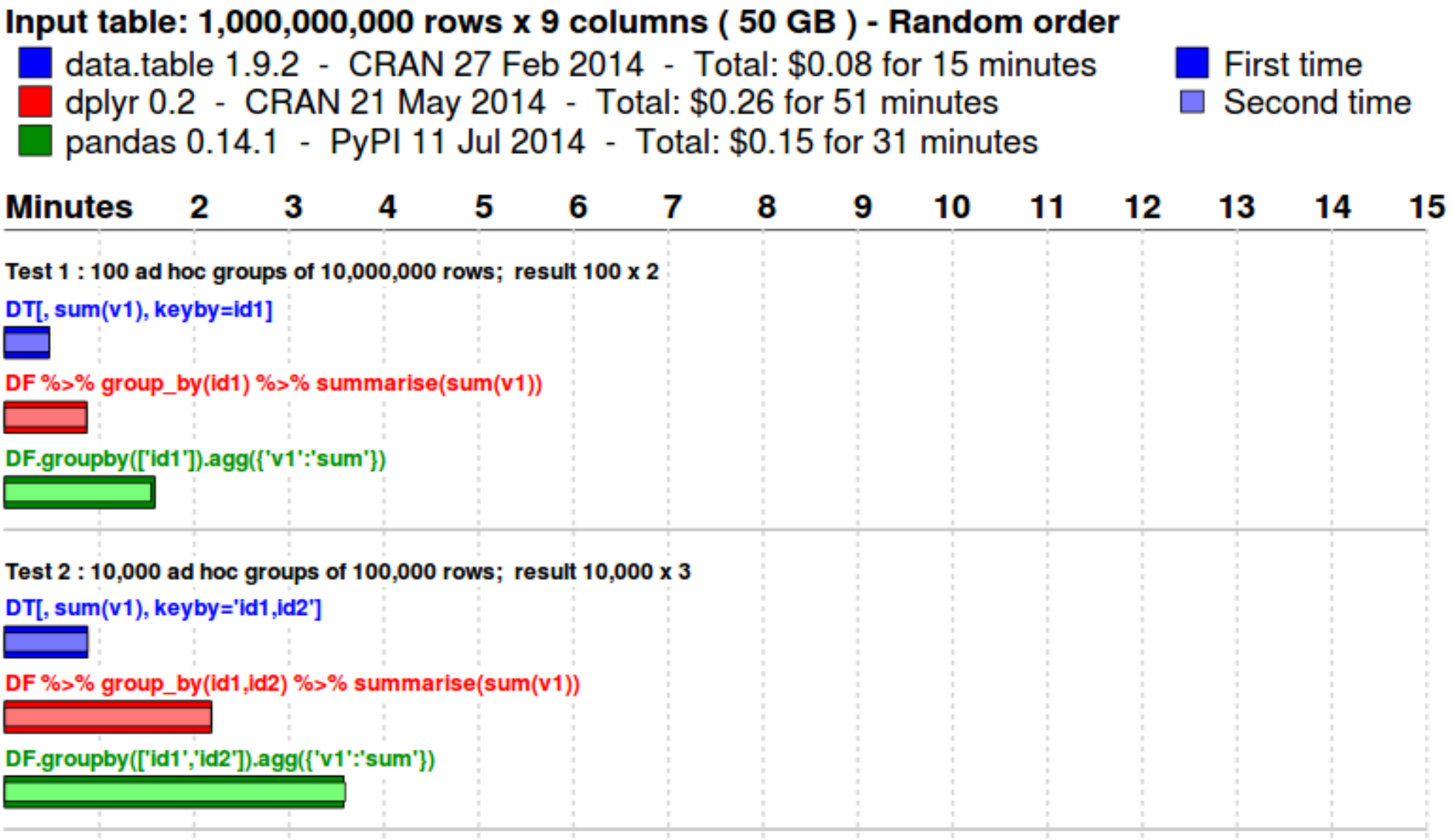
1) R's lazy evaluation enables the syntax :
   - **DT[type == ”books”, sum(sales), by=country]**
   - query optimization before evaluation
2) Pass DT to any package taking DF. It works

   is.data.frame(DT) == TRUE
3) CRAN (cross platform release, quality control)
4) Thousands of statistical packages to use with data.table

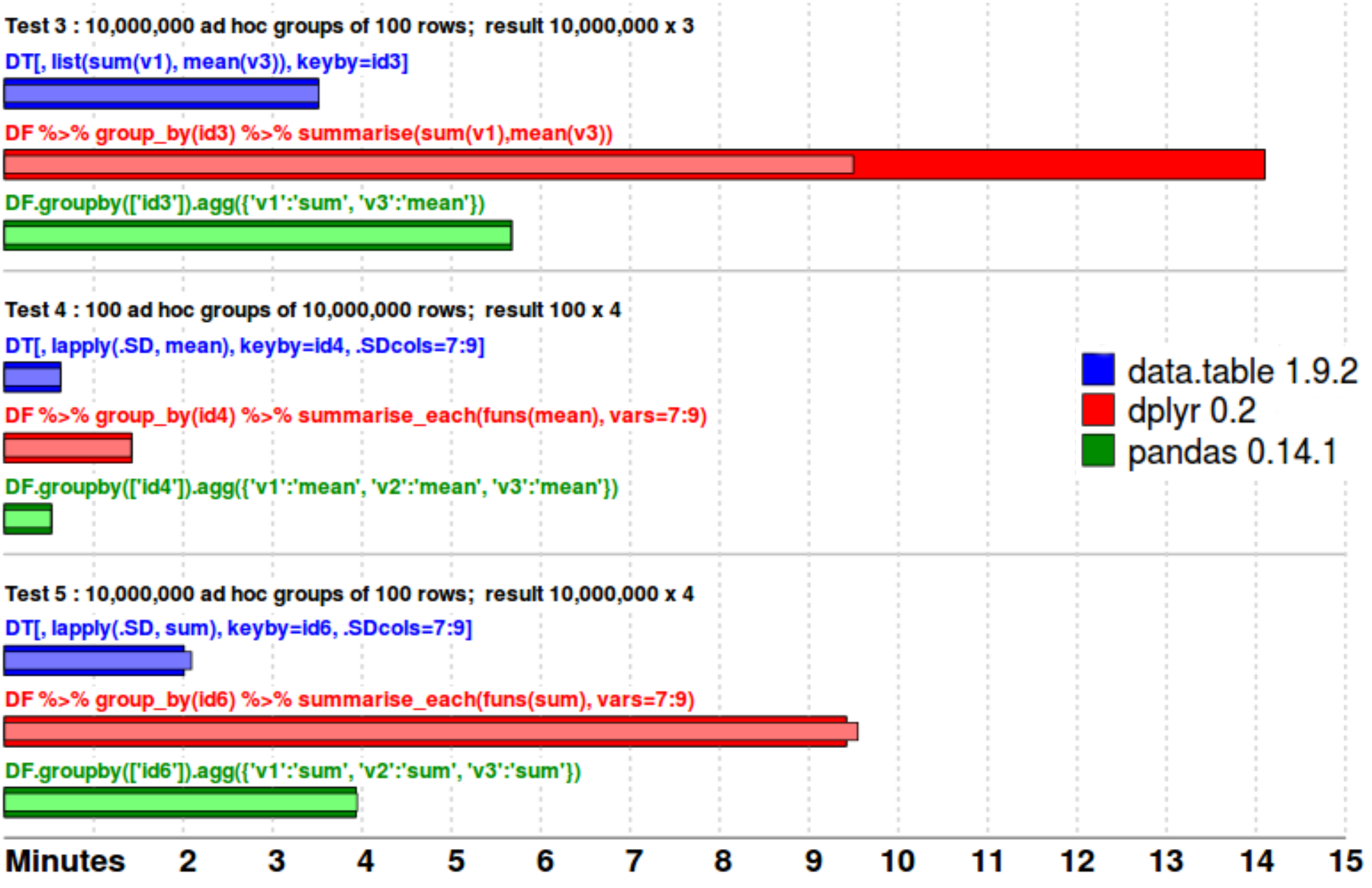
Benchmarks

- Use large data (don't loop on small data)
- Vary data
  - large groups vs small groups
  - types: integer, character, numeric, factor
  - random input or (partially) ordered
- Vary queries
  - Grouping : sum, mean, median, …
  - Joining, Updating, Subsetting
- Easy to reproduce
- Incomplete, but this is how far I've got ...

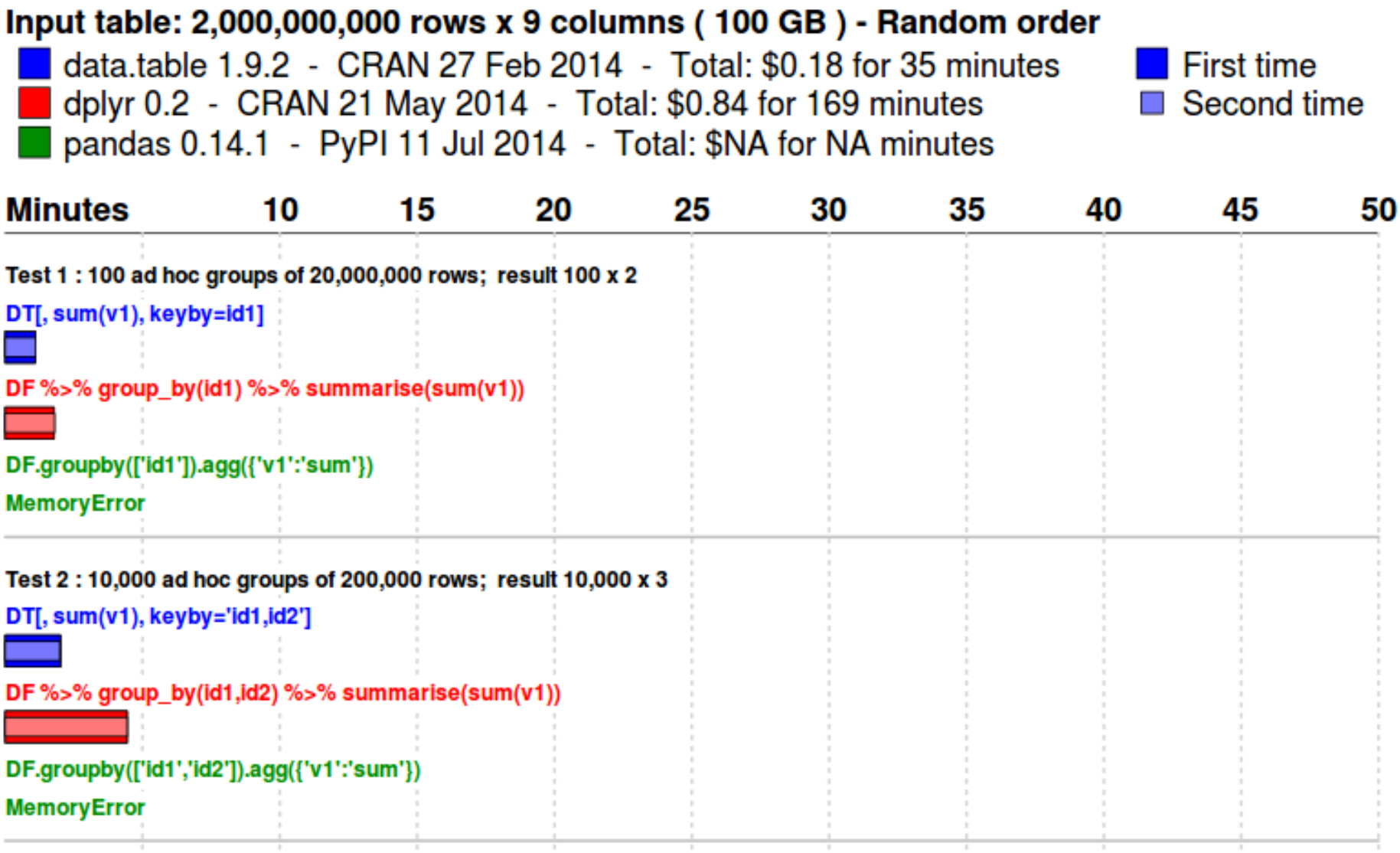
1 billion rows (50GB)



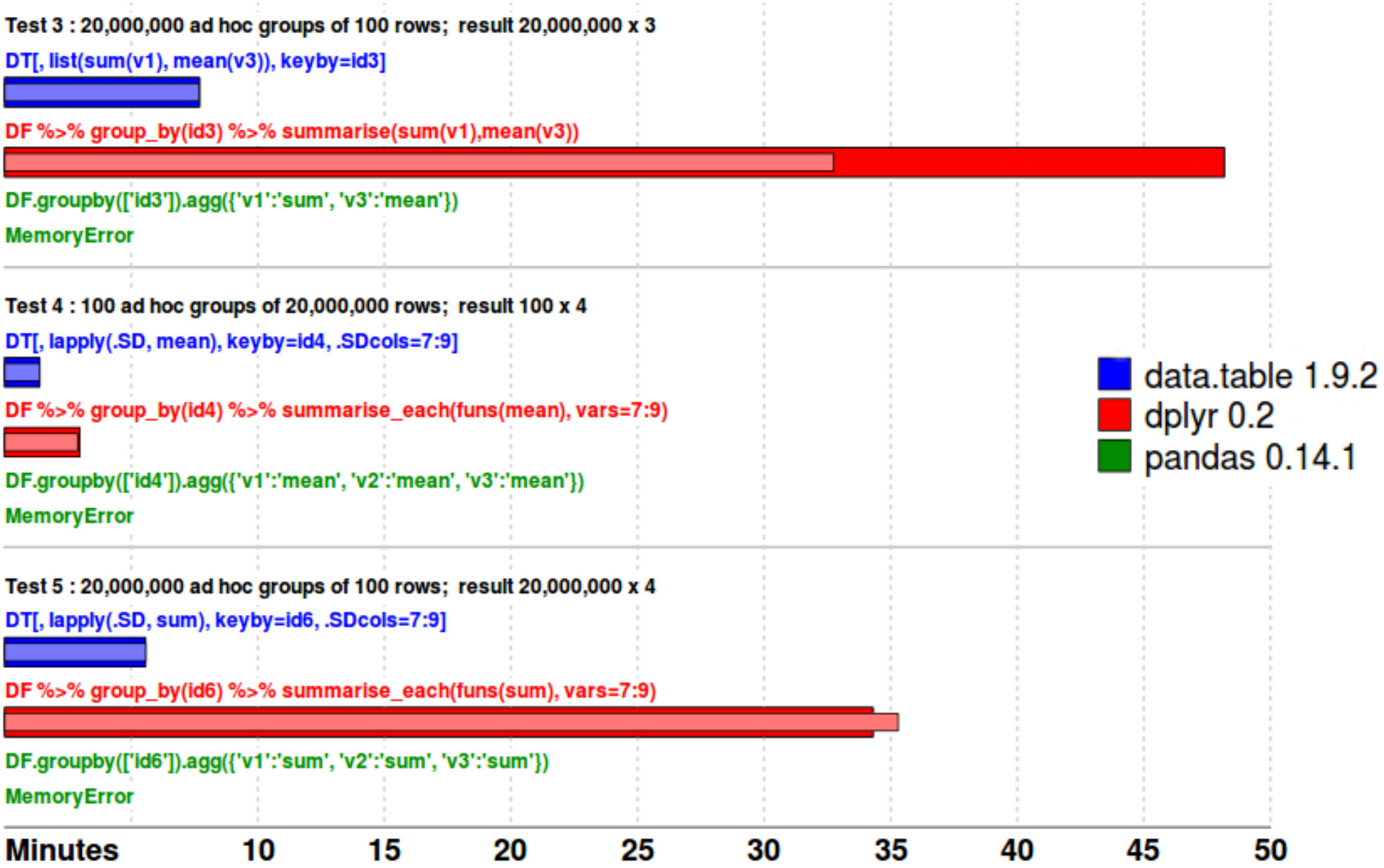
1 billion rows (50GB)



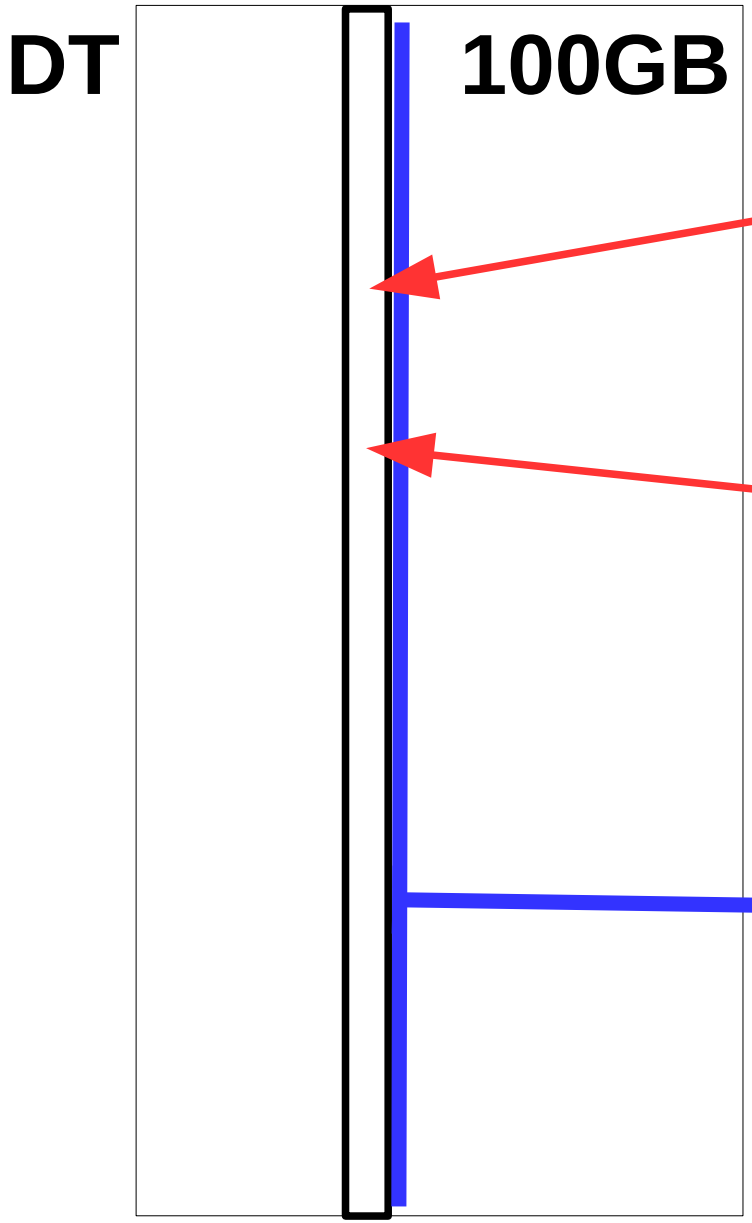
2 billion rows (100GB)



2 billion rows (100GB)



:= by reference




```
F1 = function(...) {
...
DT[ 2154634, done:=TRUE ]
...
}
```

```
F2 = function(...) {
...
DT[ 4238758, done:=TRUE ]
...
}
```

```
F3 = function(...) {
...
DT[, sum(done), by=group]
...
}
```

i.e. like a database.
We need this ability.

Client / Server Demo

Watch on YouTube (no audio)
Watch replay at EARL with Matt talking

Further reading

https://github.com/Rdatatable/data.table/wiki

http://stackoverflow.com/questions/tagged/data.table

DataCamp data.table course

3 hour data.table tutorial at useR! 2014, Los Angeles:
http://user2014.stat.ucla.edu/files/tutorial_Matt.pdf

```
> install.packages("data.table")
> require(data.table)
> ?data.table
> ?fread
```

Learn by example :

```
> example(data.table)
```