# Big Data and Supercomputing Using R

Jie Yang

Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago

October 15, 2015

# What is Big Data

- Big data is a broad term for data sets so large or complex that traditional data processing applications are inadequate (Wikipedia).
- Data is *large* if it exceeds 20% of the random access memory (RAM) on a given machine, and *massive* if it exceeds 50% (Emerson and Kane, 2012).
- Big Data represents the Information assets characterized by such a High *Volume* (amount of data), *Velocity* (speed of data in and out) and *Variety* (range of data types and sources) to require specific Technology and Analytical Methods for its transformation into value (De Mauro et al., 2015, also known as *Gartner's definition of the 3Vs*).

# Why Using R

- Handling big data requires high performance computing, which is undergoing rapid change.
- `R` is the most popular *open source* statistical software. `R` and its adds-on packages provide a wide range of high performance computing. It's free with most latest updates.
- Commercial software options (Wang, et al., 2015):
  - *RRE (Revolution R Enterprise)*: A commercialized version of R, also offers free academic use. RRE focuses on big data, large scale multiprocessor computing, and multicore functionality.
  - *SAS*: One of the most widely used commercial software for statistical analysis, provides big data support through SAS High Performance Analytics.
  - *SPSS (Statistical Product and Services Solution)*: Provide big data analytics through SPSS Modeler, SPSS Analytic Server, SPSS Analytic Catalyst (IBM, 2014), etc.
  - *MATLAB*: Provide a number of tools to tackle the challenges of big data analytics (Inc., 2014).

# Sources for R and Its Packages

- *CRAN (Comprehensive R Archive Network)*: A network of ftp and web servers around the world for R. As of 10/12/2015, it features 7333 available packages.
  https://cran.r-project.org/
- *CRAN Task Views*: Browse packages by topic and provide tools to automatically install all packages for special areas of interest. As of 10/12/2015, 33 views are available, especially *HighPerformanceComputing* for big data.
  https://cran.r-project.org/web/views/
- *Bioconductor*: Provide open source tools for the analysis and comprehension of high-throughput genomic data using R.
  https://www.bioconductor.org/
- *R Studio*: Open source and environmental software for R programming. https://www.rstudio.com/

# Windows-based R Packages for Data Management (Wang et al., 2015)

- Providing interfaces to `R` with an external database management system such as `MySQL`, `PostgreSQL`, `SQLite`, `H2`, `ODBC`, `Oracle`, etc:
  - `sqldf` (Grothendieck, 2014)
  - `RSQLite` (Wickham et al., 2014)
- Providing a simple database itself: `filehash` (Peng, 2006)
- Providing data structures for massive data while retaining a look and feel of R objects: `ff` (Adler et al., 2014)

# Windows-based R Packages for Numerical Calculation (Wang et al., 2015)

- `speedglm` (Enea, 2014): Fitting linear and generalized linear models to large data sets; computing $X'X$ and $X'y$ in increment.
- `biglm` (Lumley, 2013): Bounded memory linear and generalized linear models; computing incremental $QR$ decomposition (Miller, 1992).
- `ffbase` (Jonge et al., 2014): Basic statistical functions for Package `ff`.
- `biglars` (Seligman et al., 2011): Scalable least angle regression and LASSO.
- `PopGenome` (Pfeifer et al., 2014): An efficient "Swiss Army Knife" for population genetic and genomic analysis.

# Speeding Up: Integrate R with Others (Wang et al., 2015)

- `inline` (Sklyar et al., 2013): Wrap `C/C++` or `FORTRAN` code as strings in `R`.
- `Rcpp` (Eddelbuettel et al., 2011): Provide C++ classes for many basic `R` data types. Has been used by hundreads of other `R` packages.
- `RInside` (Eddelbuettel and Francois, 2014): Provide easy access of `R` objects from `C++`.
- `RcppArmadillo` (Eddelbuettel and Sanderson, 2014): Connect `R` with `Armadillo`, a powerful linear algebra library.

# Speeding Up: Diagnostic Tools (Wang et al., 2015)

- `microbenchmark` (Mersmann, 2014): Provide very precise timings for small pieces of source code.
- `proftools` (Tierney and Jarjour, 2013): Provide tools to analyze profiling outputs.
- `aprof` (Visser, 2014): Directed optimization for analyzing profiling outputs.
- `GUIProfiler` (de Villar and Rubio, 2014): Provide visualization of profiling results.

# Scaling Up: Parallel Computing (Wang et al., 2015)

- `Rmpi` (Yu, 2002): Provide an R interface to the Message Passing Interface (MPI) in parallel computing.
- `snow` (Rossini et al., 2007): Provide an abstract layer with the communication details hidden from the end users. Some of it has been incorporated into the based `R` package `parallel`.
- `snowFT` (Sevcikova and Rossini, 2012) and `snowfall` (Knaus, 2013): Extend `snow` with fault tolerance and wrappers for easier development of parallel `R` programs.
- `foreach` (Revolution Analytics and Weston, 2014): Allow general iterations without any explicit loop counter.

# Scaling Up: Parallel Computing (Continued)

- *Programming with Big Data in* `R` project (pbdR) (Ostrouchov et al., 2012):
  - `pbdMPI`: Provide `S4` classes to directly interface with MPI (message passing interface) to support the Single Program Multiple Data (SPMD) parallelism.
  - `pbdSLAP`: Serve as a mechanism to utilize a subset of functions of scalable dense linear algebra.
- Extra care on random number generation towards parallel computing:
  - `rlecuyer` (Sevcikova and Rossini, 2012b): Provide an interface to the random number generator with multiple independent streams.
  - `doRNG` (Gaujoux, 2014): Provide functions to perform reproducible parallel `foreach` loops.

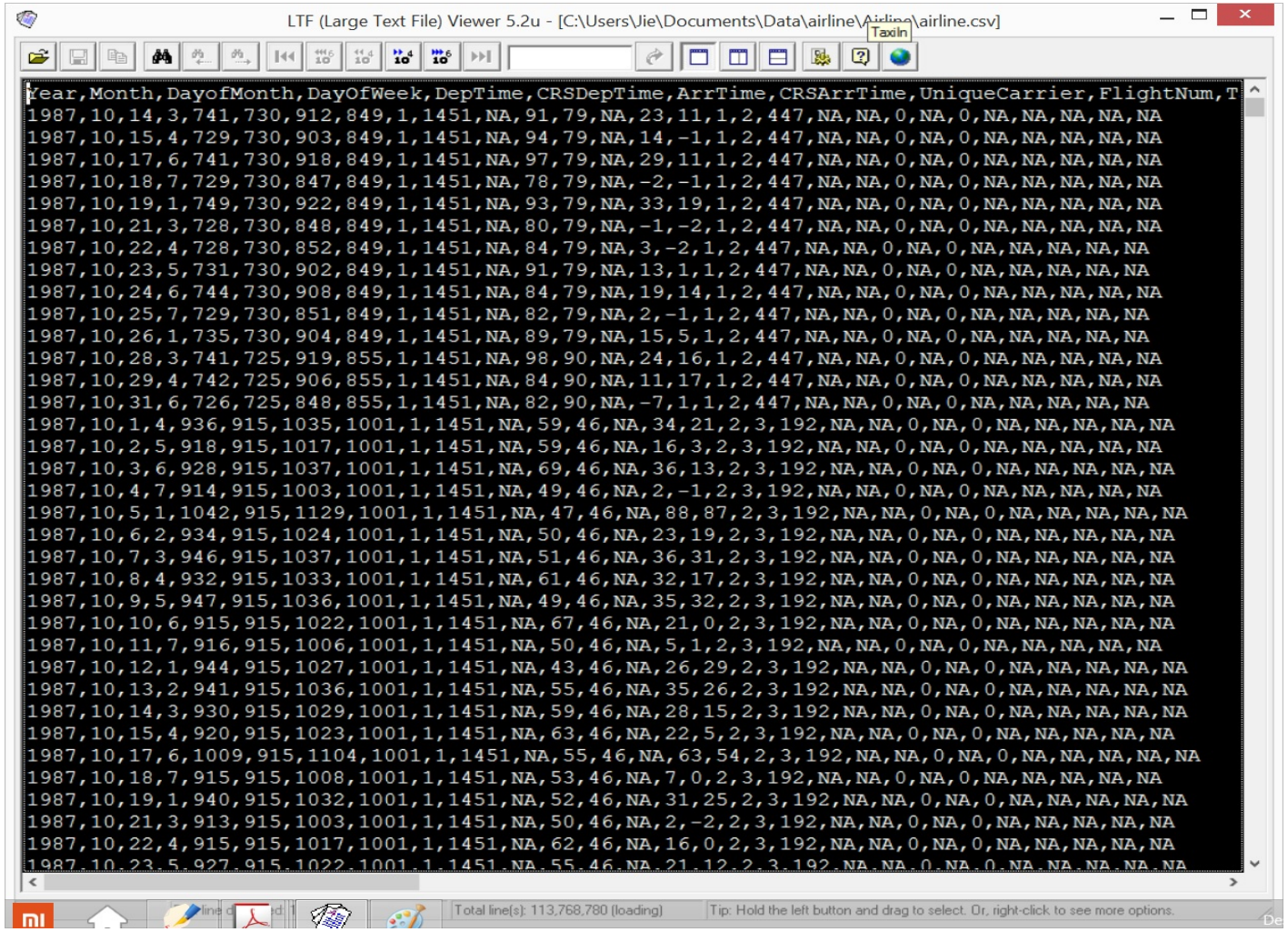
# Airline On-time Performance Data (Wang et al., 2015)

- Source: 2009 ASA Data Expo, `http://stat-computing.org/dataexpo/2009/the-data.html`; or Kane et al. (2013), `http://data.jstatsoft.org/v55/i14/Airline.tar.bz2`.
- Data: 12GB, about 120 million flights from October 1987 to April 2008, recorded with 29 variables
- Response: `Late arrival` (1 for late by more than 15 mins; 0 otherwise).
- Transformed covariates: 2 binary (`night`, `weekend`); 2 continuous (`departure hour`, `distance`).

# A Large Text File Viewer: `LTFViewer`

`LTFViewer`: A very useful tool for viewing very large text files.

`http://www.symantec.com/connect/sites/default/files/LTFViewer.zip`



# Preparation for Analyzing Big Data

- `Cygwin` (https://www.cygwin.com/): Get that `Linux` feeling on `Windows`. Also install package `R` in `Math` category and all packages in `Devel` category for `R` package `bigmemory`.
- `bzip2 -d Airline.tar.bz2` – extract the file to the current directory.
- `tar -xvf Airline.tar` – extract the file to the current directory.
- `install.packages("bigmemory")` – for `R` with Linux or Unix. Also install package `biglm`.
- `install.packages("ff")` – for `R` with Windows. Also install package `biglm`.

# Reading Airline Data into R (Kane et al., 2013)

- The data file `airline.csv` is about 12 GB.
- The use of R's native `read.csv` would require about 32 GB of RAM.
- Solution provided by `read.big.matrix` in `bigmemory`:

```
> library("bigmemory")
> ttemp=proc.time()  # on a laptop with intel(R) CPU @ 2GHz, 8GB RAM
> x <- read.big.matrix("airline.csv",
+                      header=TRUE, backingfile="airline.bin",
+                      descriptorfile="airline.desc", type="integer")
> dim(x)
[1] 123534969        29
> proc.time()-ttemp  # 2459.172
> ttemp=proc.time()
> summary(x[, "DepDelay"])
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
-1410.0    -2.0     0.0     8.2     6.0  2601.0 2302136
> proc.time()-ttemp  #  7.657
```

# Comparison among Three Strategies (Wang et al., 2015)

Time cost in secs on a 8-core machine running Linux, with Intel Core i7 2.93GHz CPU, and 16GB memory:

| | Reading data | Transforming variables | Fitting logistic regression |
|---|---|---|---|
| `bigmemory` | 968.6 | 105.5 | 1501.7 |
| `ff` | 1111.3 | 528.4 | 1988.0 |
| `RRE` | 851.7 | 107.5 | 189.4 |

The fitted logistic regression models are the same from all the three approaches:

$$\begin{aligned} & \mathrm{logit}(\text{Chance of late arrival}) \\ = \; & -2.985 + 0.104 \cdot \texttt{DepHour} + 0.235 \cdot \texttt{Distance} \\ & -0.448 \cdot \texttt{Night} - 0.177 \cdot \texttt{Weekend} \end{aligned}$$

# Super Computing: Argo Cluster at UIC

- Argo cluster (`http://accc.uic.edu/service/argo-cluster`) is a group of servers which are interconnected and used for serial or parallel program execution.
  It's free for UIC faculty, staff, and students.
- The purpose behind clustering is to make a group of computers perform as a single system and to deliver supercomputer performance.
  Argo has 57 computers connected.
- You need an Argo account and then use an SSH client to connect to argo at `argo.cc.uic.edu`.
- For example, you may first install `SecureCRT` (free at UI webstore, `https://webstore.illinois.edu`) and use it to ssh your account at `math.uic.edu` and then
  `ssh -l username argo.cc.uic.edu`.

# Install R at Argo

- Log into argo: `ssh -l username argo.cc.uic.edu`
- Get zipped R source file: `sftp username@math.uic.edu` and then `get R-2.12.2.tar.gz`
- Unzip R source file: `gzip -d R-2.12.2.tar.gz` and then `tar -xf R-2.12.2.tar`
- Install R: `cd R-2.12.2`, then issue commands (at the shell prompt) `./configure` and then `make`.
- Run R at Argo: `/home/homes53/username/R/R-2.12.2/bin/R`
- Install R packages: `install.packages( "/home/homes53/username/R/e1071.tar.gz", lib="/home/homes53/username/R", repos = NULL)` or `install.packages("ff", lib="/home/homes53/username/R", repos = "http://cran.us.r-project.org", dependencies =T)`

# An Example of R Batch Job: test1.r

```
## /usr/common/R/bin/R CMD BATCH
##        /home/homes53/username/test/test1.r testout &
## need data sets "y100.dat"
Y100 <- read.table("/home/homes53/username/test/y100.dat")
idata <- 1                    # first simulated data
Y <- as.vector(as.matrix(Y100[idata,1:20]))

sink("test1out.txt")          # save output into a file
cat("\n Data set:")
print(round(Y,3))

x <- rnorm(100)
cat("\n max=", max(x),"\n")
sink()
save.image(file="/home/homes53/username/test/test1.RData")
```

# Submit an R Script File to Argo

- Compile your script file on master node, for example, "script1":

```
#PBS -m bea
#PBS -e /home/homes53/username/test/test1.error
#PBS -o /home/homes53/username/test/test1.output
#PBS -N test1
/home/homes53/username/R/R-2.12.2/bin/R CMD BATCH
     /home/homes53/username/test/test1.r
```

- Submit your script to Argo: `qsub script1`
- See which nodes are currently in use: `qnodes | more`
- Submit a job to a particular node:
  `qsub -l nodes=argo1-1 script1`

# Monitor Your Jobs at Argo

- Check the status of all your jobs: `qstat`
- Check the status of a particular job: `qstat <job number>`
- Check the status of all running jobs: `qstat -a` or `qstat -an`
- Check if job 20836 is running. If not, then the reason it is not running is given: `qstat -s 20836`
- Cancel job 20836: `qdel 20836`

# Reference


- Emerson, J.W. and Kane, M.J. (2012). Don't drown in the data. *Significance*, **9**, 38–39.
- De Mauro, A., Greco, M., and Grimaldi, M. (2015). What is big data? A consensual definition and a review of key research topics. *AIP Conference Proceedings*, **1644**, 97–104.
- Kane, M.J., Emerson, J., and Weston, S. (2013). Scalable strategies for computing with massive data. *Journal of Statistical Software*, **55**, 1-19.
- Wang, C, Chen, M.-H., Schifano, E., Wu, J., and Yan, J. (2015). A Survey of Statistical Methods and Computing for Big Data, http://arXiv:1502.07989v1