R with HPC

Burak Himmetoglu
ETS & CSC
bhimmetoglu@ucsb.edu

11/03/2016

Things you can do with R (+RStudio)

Computational (base R + libraries)

- Statistics
- Simulations
- Data Analysis
- Predictive analysis, machine learning
- …

All other cool stuff (RStudio)

- Projects (R packages, Github, …)
- Web pages, presentations (Knitr, …)
- Web applications (Shiny,…)
- …

Question: My R solution is taking too long on my computer, what can I do?

Possible answers:

- Use specialized packages for performance 😀
- Try simple (shared memory) parallel tools 😀
- Run your R code in a remote cluster 😀/😐
  - Large datasets that don't fit your computer's memory
  - Divide and Conquer

- Try (distributed memory) parallelism, or Hadoop solutions 😬/😱
- Write C/C++ extensions for R 😬/😱

R on a remote computer cluster

Possible drawbacks

- Almost all computer clusters run Linux 😎/😩
- You can't use RStudio interactively on a cluster 😩
- Need to be submitted to a queuing system 😕

Advantages

- Access to a large memory (> 40 GB, up to 1 TB)
- Fire and forget: Submit many calculations!

Examples in this seminar

If you have an account on Knot:

`export PATH="/sw/csc/R-3.2.3/bin:$PATH"`

Download the exercises from the command line:

`svn checkout https://github.com/bhimmetoglu/talks-and-lectures/trunk/CSC-UCSB`

All exercises are online:

https://github.com/bhimmetoglu/talks-and-lectures

Example 1: Run R code on Knot cluster

montecarlo.R

Monte Carlo integration:

$$Z = \int_0^1 \int_0^1 \ldots \int_0^1 dx_1\, dx_2\, \ldots dx_n\, e^{-x_1^2 - x_2^2 - \cdots - x_n^2}$$

For (i = 1, NumSimulations){

Pick $\{x_1, x_2, \ldots, x_n\}$ from a uniform distribution

Z ← (Volume of region) x (Integrand at $\{x_1, x_2, \ldots, x_n\}$)

}

Average results (Z's)

Example 1: Run R code on Knot cluster

- Remember: No RStudio to experiment with!
- Make sure that your R code runs from start to end
- Perform tests on your computer first

A simple script (text file) can be used to submit to the queue:

```
#!/bin/bash
#PBS -l nodes=1:ppn=12
#PBS -l walltime=01:00:00
#PBS -N MonteCarlo
#PBS -V

cd $PBS_O_WORKDIR

Rscript --vanilla montecarlo.R > output
```

Example 1: Run R code on Knot cluster

Let's say that the name of the script is: `submit.job`

```
qsub submit.job
```

Better use the short queue, since this is a test job < 1 hr

```
qsub -q short submit.job
```

Check status:

```
showq -u $USER
```

Example 2: Titanic Survival Prediction

https://www.kaggle.com/c/titanic

Jack:

$P(Survived) \simeq 0.19$

Rose:

$P(Survived) \simeq 0.74$

Prediction purely based on gender

Example 2: Titanic Survival Prediction

This example illustrates:

Importance of using R packages that replace base R functions

e.g.: dplyr, readr

- Written in C++, fast, easy to use
- Use them on cluster and on your computer

Importance of using parallel capabilities already encoded

e.g. : glmnet

- Shared memory parallelizm implemented
- Take advantage of it!

Why dplyr?

Proportion of survived passengers by gender

Base R way:

```
P_Surv_M <- sum(train[train$Sex == "male",]$Survived)/length(train[train$Sex == "male"])

P_Surv_F <- sum(train[train$Sex == "female",]$Survived)/length(train[train$Sex == "female"])
```

dplyr way:

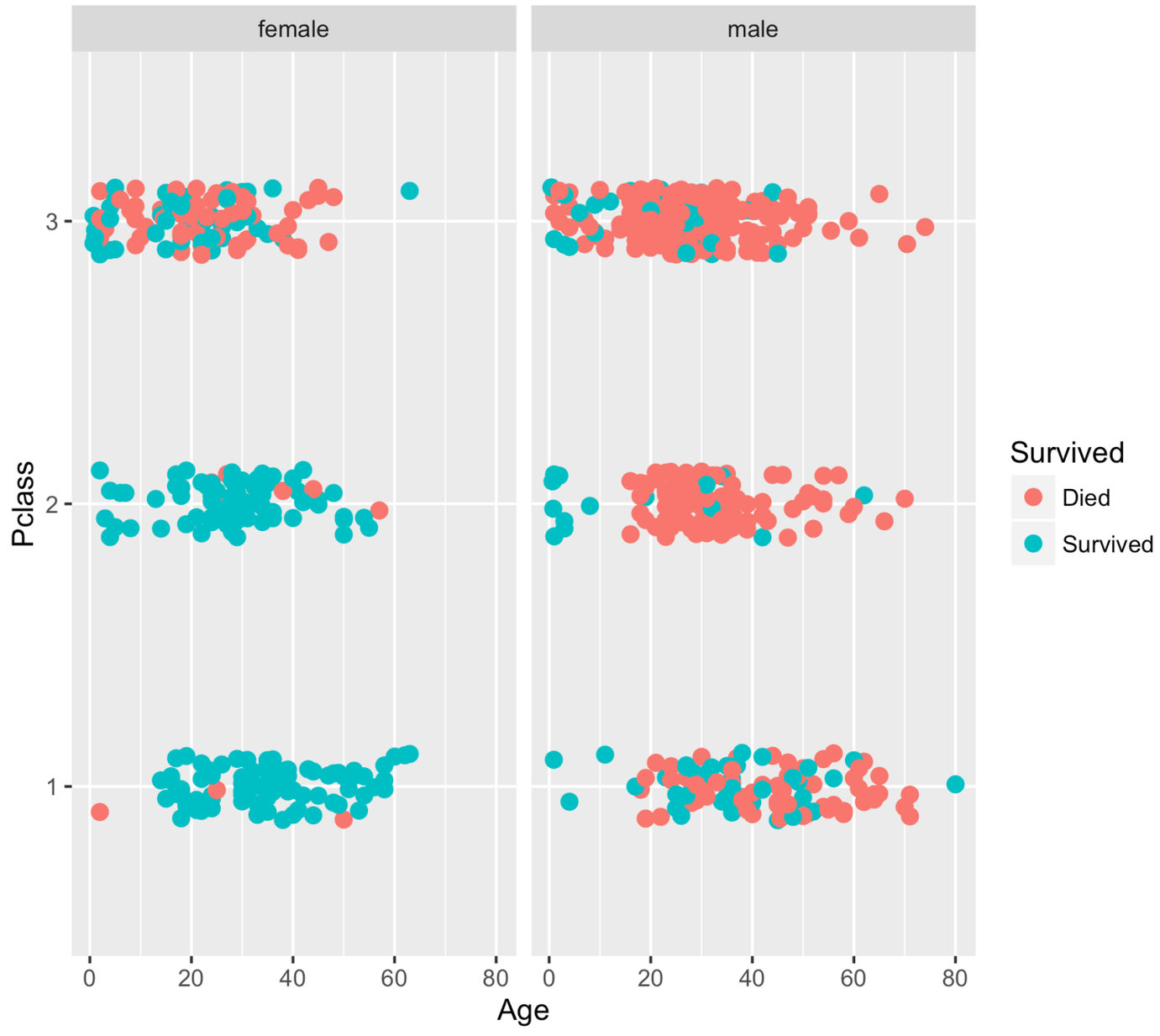
```
train %>% group_by(Sex) %>% summarize(survivalRate = sum(Survived == TRUE)/n() )
```

- Intuitive
- Efficient and fast
- Less memory
- Complicated chain of tasks in a simple line of code

Why dplyr?

```
> train %>% group_by(Sex,Pclass) %>% summarize(n())
Source: local data frame [6 x 3]
Groups: Sex [?]

     Sex Pclass   n()
   (chr)  (int) (int)
1 female      1    94
2 female      2    76
3 female      3   144
4   male      1   122
5   male      2   108
6   male      3   347
```

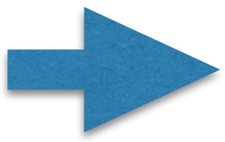


Model Matrices

- We need to convert all factor variables into numeric ones
- In general, values cannot be compared
- E.g. States in U.S, Gender, City etc.

`model.matrix()`
`sparse.model.matrix()`

Id	Pclass	Age
1	1	45
2	2	50
3	2	22
4	3	18
5	1	65
6	2	34



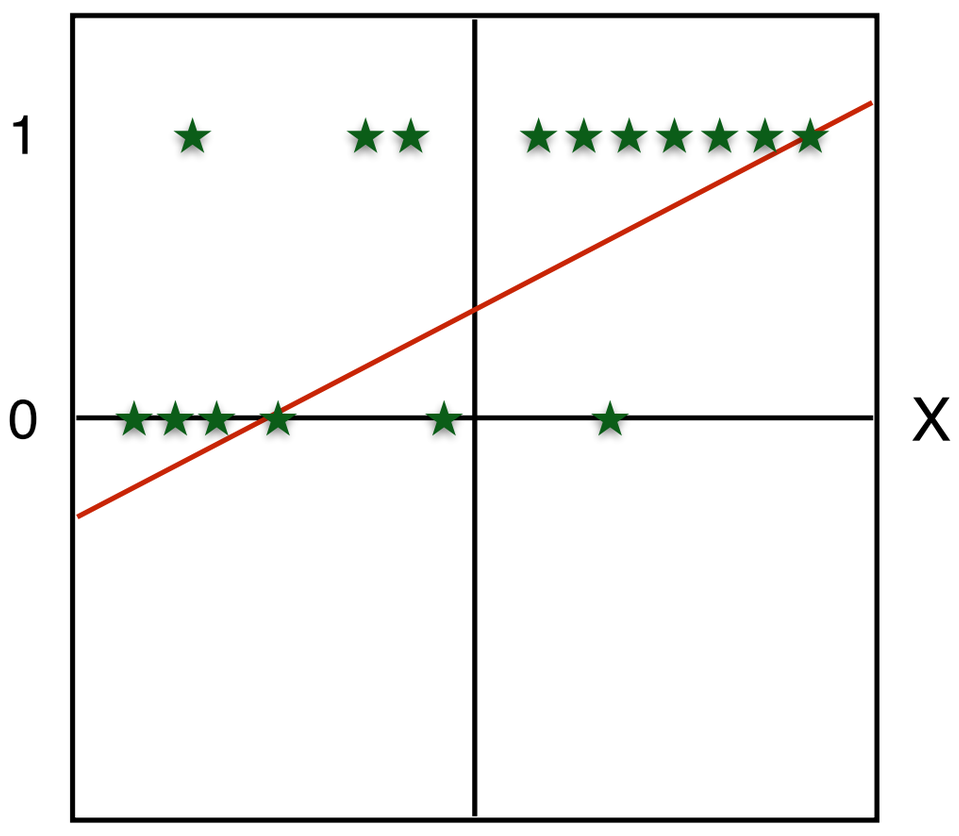
Id	Pclass2	Pcalss3	Age
1	0	0	45
2	1	0	50
3	1	0	22
4	0	1	18
5	0	0	65
6	1	0	34

Logistic Regression

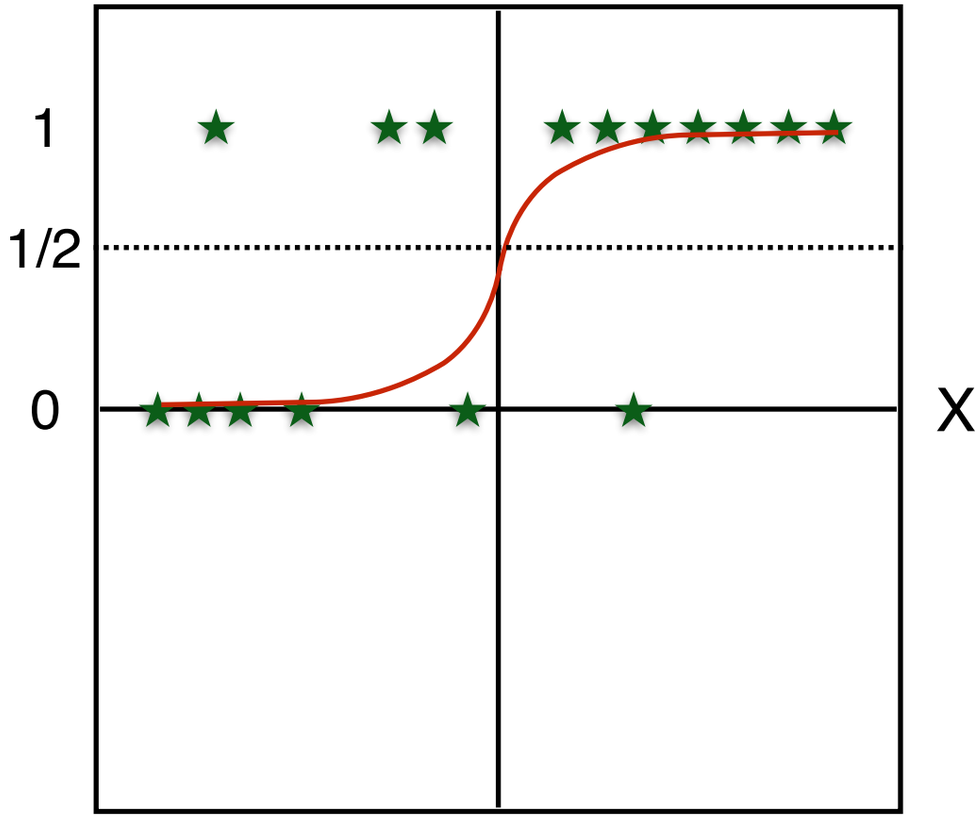
- Linear model for classification

$$z_i = \beta_0 + \beta_1^T \cdot \mathbf{x}_i \qquad y_{\text{pred, i}} = \frac{1}{1 + e^{-z_i}}$$

Survived



Survived



Logistic Regression (Regularization)

- Parameters β_0, β_1 optimized to yield small error
- Overfitting problem: LASSO and Ridge regression
- α, λ by cross-validation (parallel part in glmnet)

This is the optimization problem:

$$\min_{\beta_0,\beta} \frac{1}{N} \sum_{i=1}^{N} l(y_i, \beta_0 + \beta^T x_i) + \lambda \left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1 \right]$$

```
# Functions to use:
cv.glmnet() # Determines  λ  by cross-validation
glmnet()    # Determines  β0, β1  by optimization
```

Example 3: Flip coins and aggregate results

`foreach` package

- Testing parallel performance in a simple experiment
- Flip 100 coins for 10,000 times, store results in a table
- Look at the script `coinFlips.R`

```
nSim = 1e+5; nSpin = 100

coinFlips <- matrix(0, nrow = nSim, ncol = nSpin)
coinFlips <- foreach(i=1:nSim, .combine = rbind) %dopar%
  (rbinom(n = nSpin, size = 1, prob = 0.5))

coinFlips <- matrix(0, nrow = nSim, ncol = nSpin)
coinFlips <- foreach(i=1:nSim, .combine = rbind) %do%
  (rbinom(n = nSpin, size = 1, prob = 0.5))
```

More Exercises

Compare the timing in the scripts:

1. montecarlo.R
2. montecarloSer.R
3. montecarloPar.R

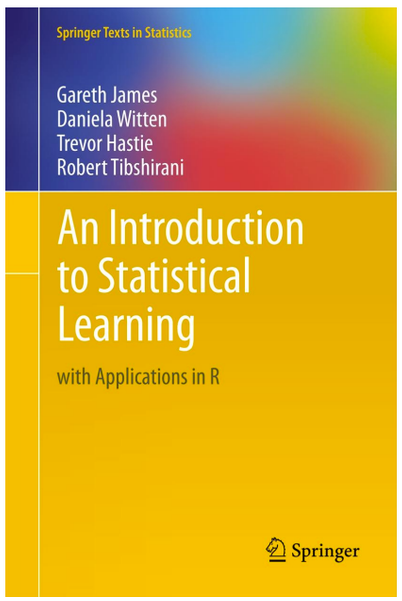
Which one runs the fastest and why?

Resources for learning R

- swirl package
- Coursera : https://www.coursera.org/learn/r-programming
- LeanPub: https://leanpub.com/rprogramming
- Lynda : Up and Running with R

Introduction to Statistical Learning with applications in R

http://www-bcf.usc.edu/~gareth/ISL/



Resources for high performance computing with R

Packages:

- Rcpp
- data.table
- snow, Rmpi
- H2O
- RHadoop
- Rhipe
- …

https://cran.r-project.org/web/views/HighPerformanceComputing.html