Analyzing Analytics: Advanced Performance Analysis Tools for R

Drew Schmidt*, Wei-Chen Chen, Christian Heckendorf, George Ostrouchov

November 14, 2016

The pbdR Core Team

Wei-Chen Chen[1]
George Ostrouchov[2,3]
Drew Schmidt[3]

Support

This material is based upon work supported by the National Science Foundation Division of Mathematical Sciences under Grant No. 1418195.

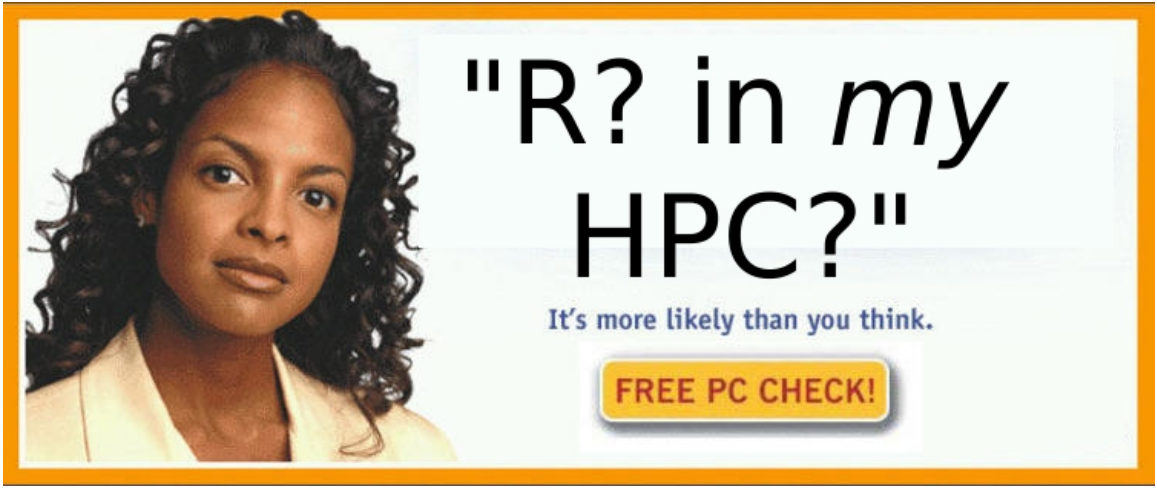
The findings and conclusions in this presentation have not been formally disseminated by the U.S. Department of Health & Human Services nor by the U.S. Department of Energy, and should not be construed to represent any determination or policy of University, Agency, Administration and National Laboratory.

[1]FDA
Washington, DC, USA

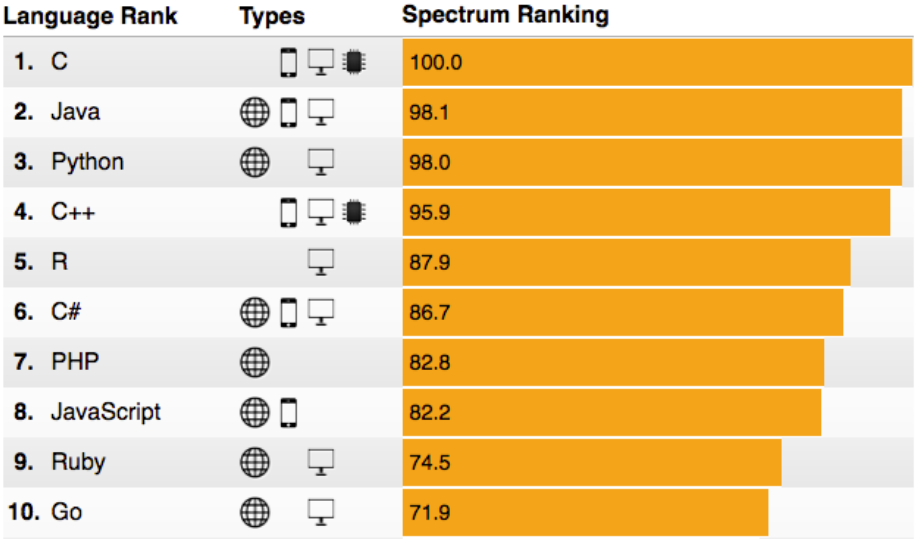
[2]Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge TN, USA

[3]University of Tennessee
Knoxville TN, USA

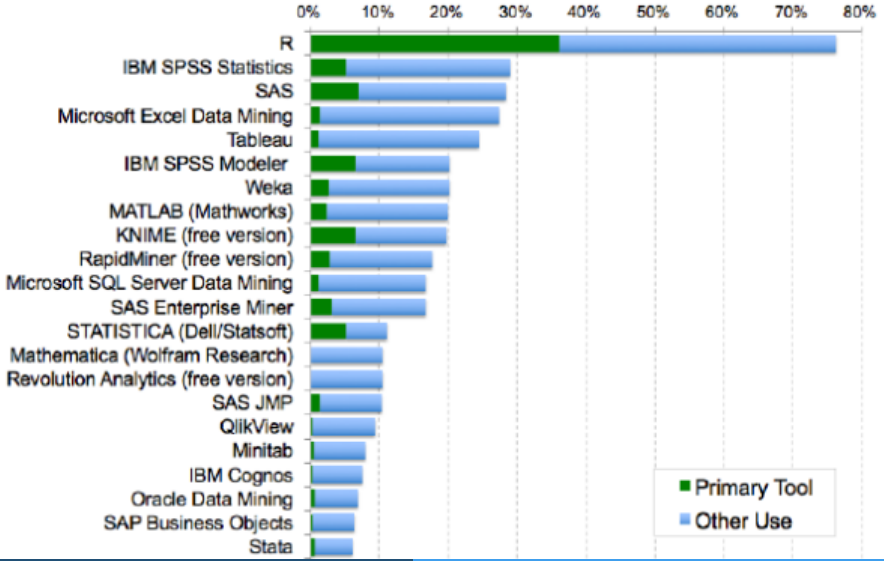
Contents

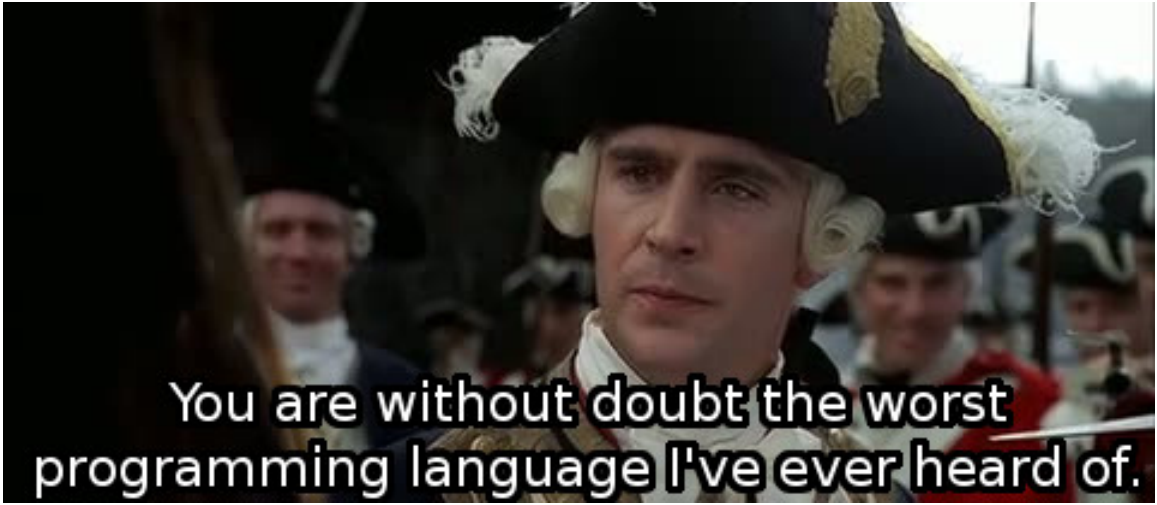

"R? in my HPC?"
It's more likely than you think.
FREE PC CHECK!

IEEE Spectrum's 2016 Ranking of Programming Languages



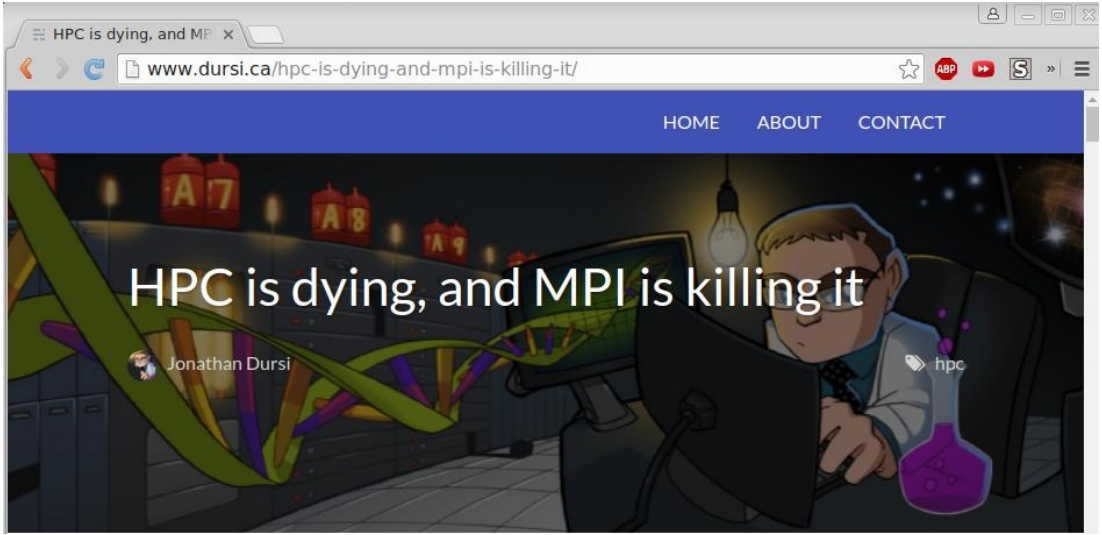
Rexer 2015 data scientist survey



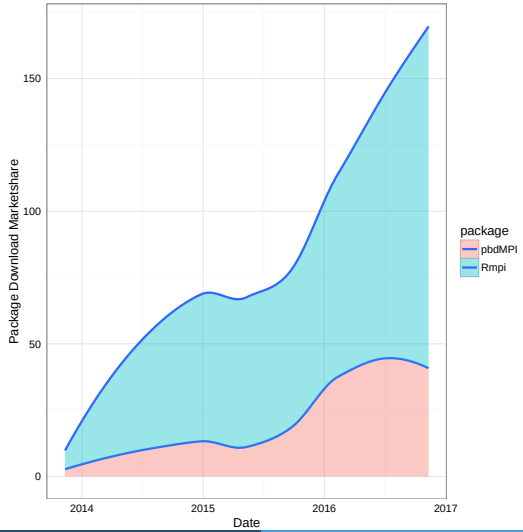

You are without doubt the worst
programming language I've ever heard of.


But you HAVE heard of me!

About Traditional HPC...



HPC may be dying, but we're behind the times



"OLCF Researchers Scale R to Tackle Big Science Data Sets"

- A problem that takes several hours on Apache Spark [was analyzed] in less than a minute using R on OLCF high-performance hardware.
- "*. . . for situations where one needs interactive near-real-time analysis,* **the pbdR approach is much better**."

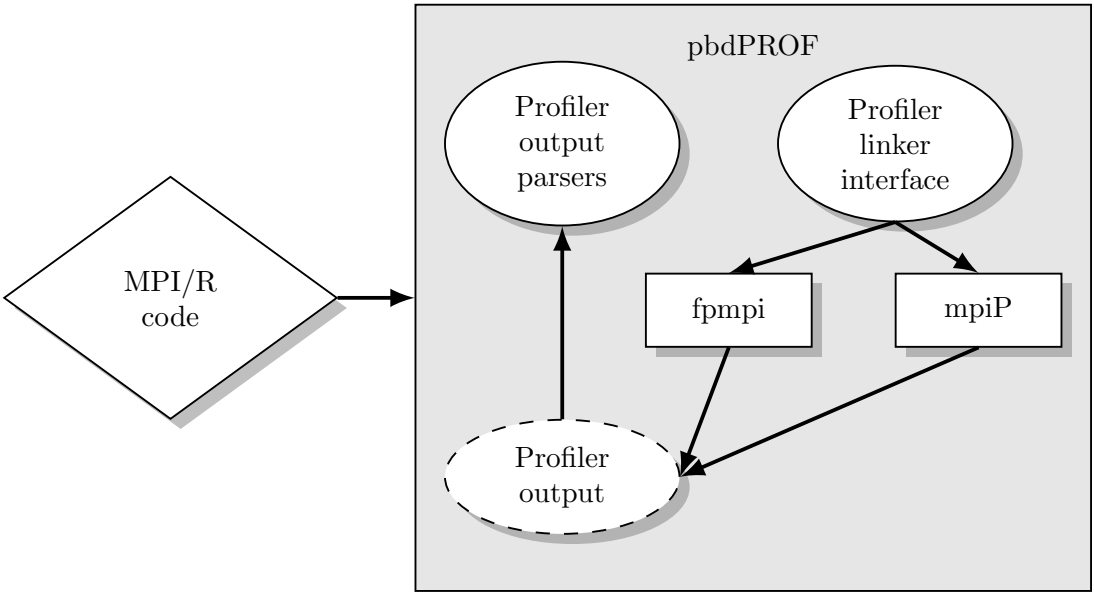
https://www.hpcwire.com/2016/07/06/olcf-researchers-scale-r-tackle-big-science-data-sets/

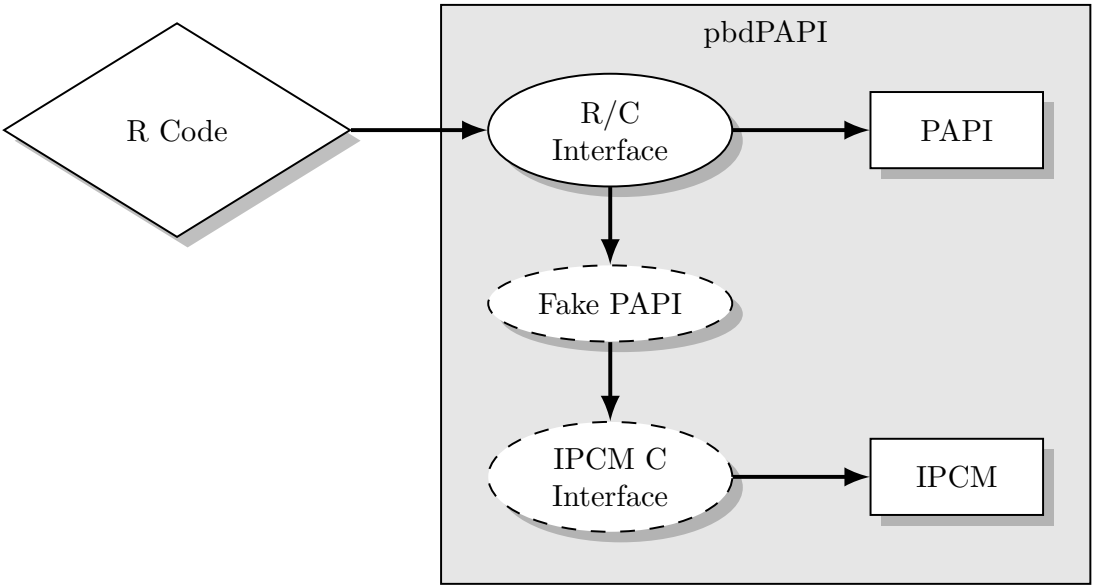
Profiling in R Versus Traditional HPC

- Profiling in R (core): Stack sampler, some memory profilers.
- Every (other) profiling package in R: wall clock times.
- What's missing: MPI profilers, performance counters.

Three New R Packages

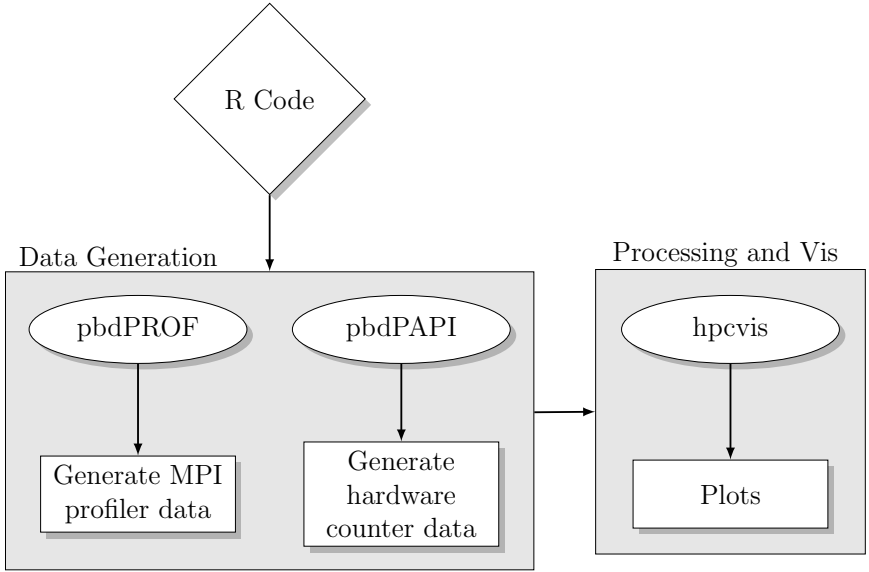
- pbdPROF - MPI profiler support (fpmpi and mpiP).
- pbdPAPI - PAPI bindings and utilities.
- hpcvis - Plot methods for the above 2.


pbdPROF
Profiler output parsers
Profiler linker interface
MPI/R code
fpmpi
mpiP
Profiler output


R Code
pbdPAPI
R/C
Interface
PAPI
Fake PAPI
IPCM C
Interface
IPCM

pbdPAPI High-Level Interface

Function	Measurement
`system.cache()`	Cache misses, hits, accesses
`system.flops()`	MFLOPS
`system.idle()`	Idle cycles
`system.cpuormem()`	Classify CPU/RAM bound
`system.utilization()`	CPU utilization


R Code
Data Generation
pbdPROF
pbdPAPI
Generate MPI profiler data
Generate hardware counter data
Processing and Vis
hpcvis
Plots

pbdPROF + hpcvis Syntax Example

```
library(pbdPROF)
library(hpcvis)

prof_file <- "R.4.26802.1.mpiP"
x <- read.prof(prof_file)

profplot(x, color=FALSE, stacked=FALSE, title=NULL, which=c(2,4))
profplot(x, plot.type="stats", color=FALSE, stacked=TRUE, title=NULL)
```

pbdPAPI + hpcvis Syntax Example

```
library(pbdPAPI)
library(hpcvis)

cb <- cachebench(foo(x), bar(x))
papiplot(cb, label.angle=15, levels=1:2)
```

Note

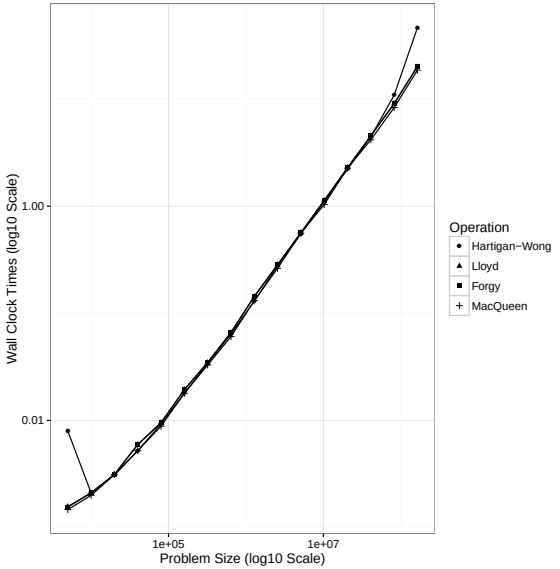
All plots for the remainder of this presentation were made with hpcvis.

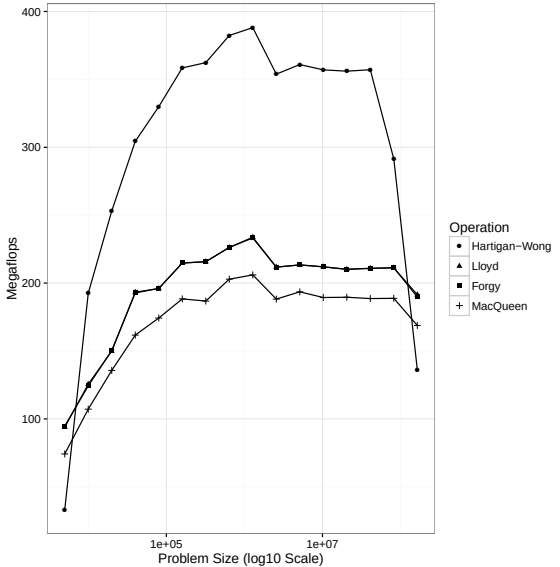
Clustering Approaches

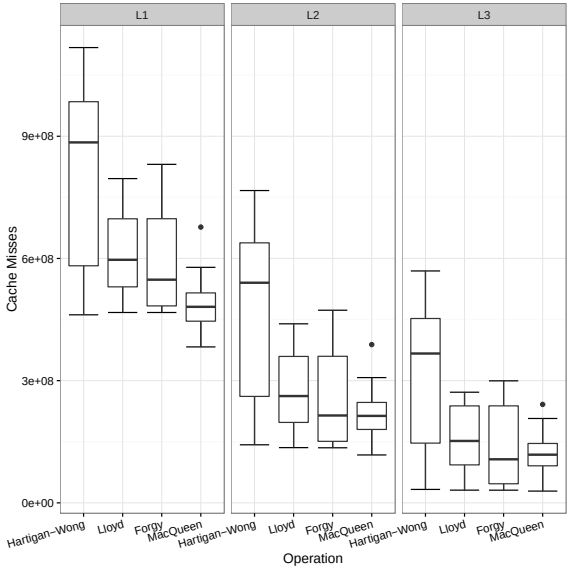
- K-means and GMM
- Two unsupervised learning approaches.
- Trying to discover latent groupings in the data.
- K-means
  - Cover data with k *identical spheres*.
- GMM
  - Can use different ellipsoids, pointing in different directions.

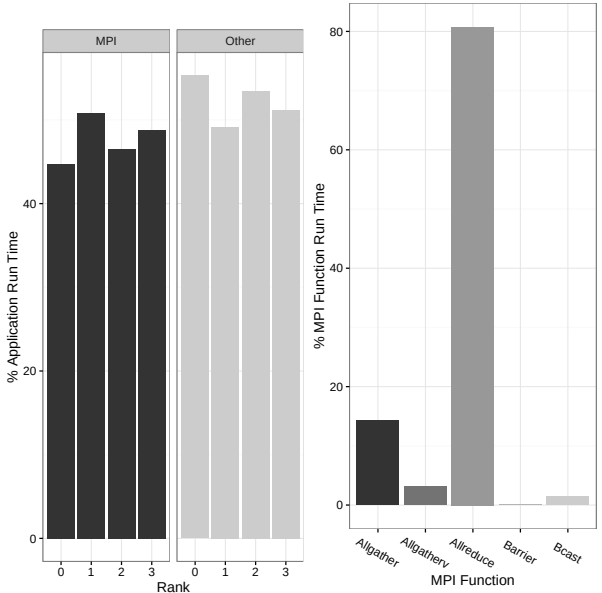
Now With Math

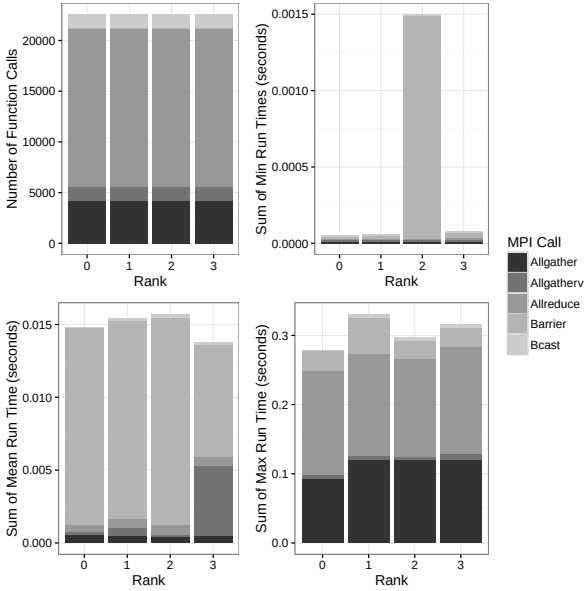
- Partition N data points in p dimensions into K clusters
- k-means model (stated as Gaussian mixture):
  - $X_1, \ldots, X_N \overset{iid}{\sim} \phi(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$
    - $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_k\}$
    - $\boldsymbol{\Sigma} = \{\sigma \boldsymbol{I}, \ldots, \sigma \boldsymbol{I}\}$
    - $\phi(\cdot|\boldsymbol{\mu}, \sigma \boldsymbol{I}) = \sum_{k=1}^{K} \eta_k \phi_p(\cdot|\mu_k, \sigma \boldsymbol{I})$
    - $2Kp + 1$ parameters
- **pmclust**: Gaussian mixture model (GMM):
  - $X_1, \ldots, X_N \overset{iid}{\sim} \phi(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$
    - $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_k\}$
    - $\boldsymbol{\Sigma} = \{\Sigma_1, \ldots, \Sigma_K\}$
    - $\phi(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \eta_k \phi_p(\cdot|\mu_k, \Sigma_k)$
    - $2Kp + K\frac{p \times (p+1)}{2}$ parameters


Wall Clock Times (log10 Scale)
1.00
0.01
1e+05
1e+07
Problem Size (log10 Scale)
Operation
Hartigan−Wong
Lloyd
Forgy
MacQueen


400
300
200
100
Megaflops
1e+05
1e+07
Problem Size (log10 Scale)
Operation
Hartigan−Wong
Lloyd
Forgy
MacQueen


L1
L2
L3
Cache Misses
9e+08
6e+08
3e+08
0e+00
Hartigan–Wong
Lloyd
Forgy
MacQueen
Operation


MPI
Other
% Application Run Time
0
20
40
0 1 2 3
0 1 2 3
Rank
% MPI Function Run Time
0
20
40
60
80
Allgather
Allgatherv
Allreduce
Barrier
Bcast
MPI Function


Number of Function Calls
20000
15000
10000
5000
0
0
1
2
3
Rank
Sum of Min Run Times (seconds)
0.0015
0.0010
0.0005
0.0000
0
1
2
3
Rank
MPI Call
Allgather
Allgatherv
Allreduce
Barrier
Bcast
Sum of Mean Run Time (seconds)
0.015
0.010
0.005
0.000
0
1
2
3
Rank
Sum of Max Run Time (seconds)
0.3
0.2
0.1
0.0
0
1
2
3
Rank

Challenges

- pbdPAPI still difficult for R users to install.
- Mostly Linux-only.

Future Work

- Build a web interface?
- Integrate with tau?

∼Thanks!∼

Questions?


Email: wrathematics@gmail.com


GitHub: https://github.com/wrathematics


Web: http://wrathematics.info


Twitter: @wrathematics