The pbdR Project: Distributed Computing with R*

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

[1]U.S. Food and Drug Administration
[2]Oak Ridge National Laboratory
[3]pbdR Core Team
RBigData@gmail.com

JSM2018, Vancouver, BC, Canada

*The pbdR project was supported in part by NSF and DOE grants

Disclaimer

Any opinions, findings, and conclusions or recommendations expressed in this presentation are those of the authors.

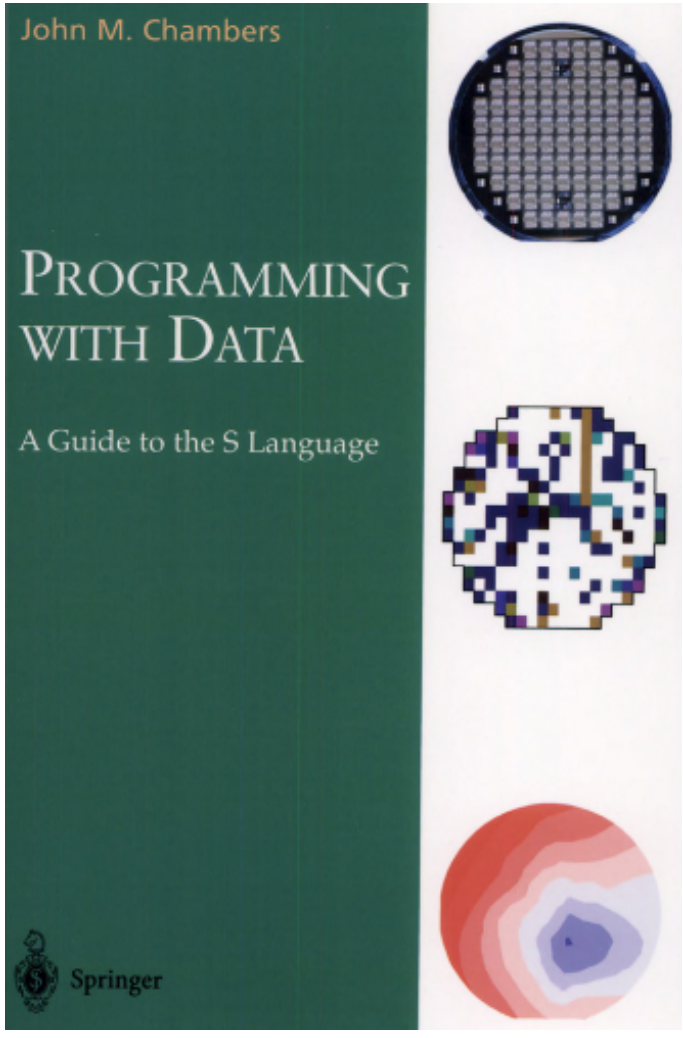
Nothing in this content has been formally disseminated by the U.S. Department of Health & Human Services or by U.S. Food and Drug Administration, and should not be construed to represent any determination or policy of University, Agency, Administration, or National Laboratory.

Outline

What is pbdR

- Around 1998, a book "*Programming with Data – A Guide to the S Language*" by John M. Chambers was published.
- Around 2012, the "**P**rogramming with **B**ig **D**ata in **R**" project was started with a set of highly scalable R packages for distributed computing and profiling in data science.

pbdR
Programming with Big Data in R



Where to learn pbdR

- Main website:
  - http://pbdr.org/
- Main release:
  - http://pbdr.org/release/
- Main repository:
  - https://github.com/RBigData/
- Where to start:
  - pbdDEMO vignettes (about 150 pages) at `https://cran.r-project.org/web/packages/pbdDEMO/vignettes/pbdDEMO-guide.pdf`
  - pbdMPI vignettes (about 30 pages) at `https://cran.r-project.org/web/packages/pbdMPI/vignettes/pbdMPI-guide.pdf`

How can pbdR help?

- Performance, performance, performance ...

Parallel pbdR

```
1  randSVD <- function(A, k, q=3)
2    {
3      ## Stage A
4      Omega <- ddmatrix("rnorm", nrow=n, ncol=2*k)
5      Y <- A %*% Omega
6      Q <- qr.Q(qr(Y))
7      At <- t(A)
8      for(i in 1:q)
9        {
10         Y <- At %*% Q
11         Q <- qr.Q(qr(Y))
12         Y <- A %*% Q
13         Q <- qr.Q(qr(Y))
14       }
15
16     ## Stage B
17     B <- t(Q) %*% A
18     U <- La.svd(B)$u
19     U <- Q %*% U
20     U[, 1:k]
21   }
```

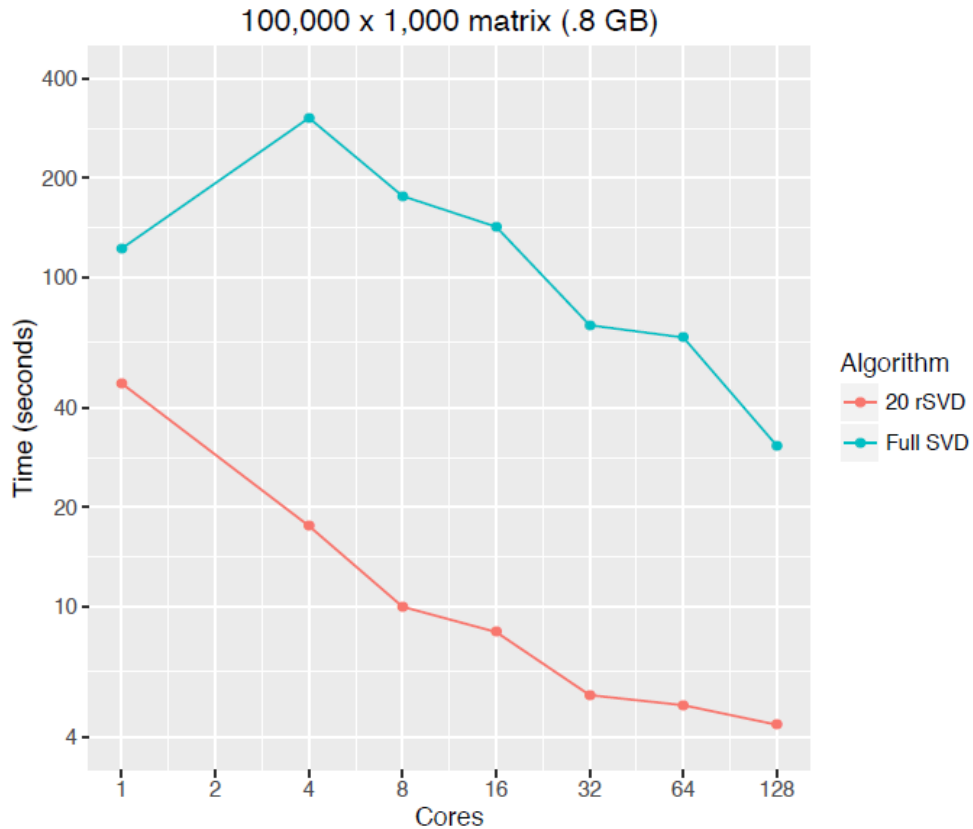


Figure 1: Parallelizing the function by one line. Execute the function in any # of cores. Obtain the results faster.

Illustration of Parallelism

- Popular for statistical simulations
- Efficient for small size of data or dividable data/tasks

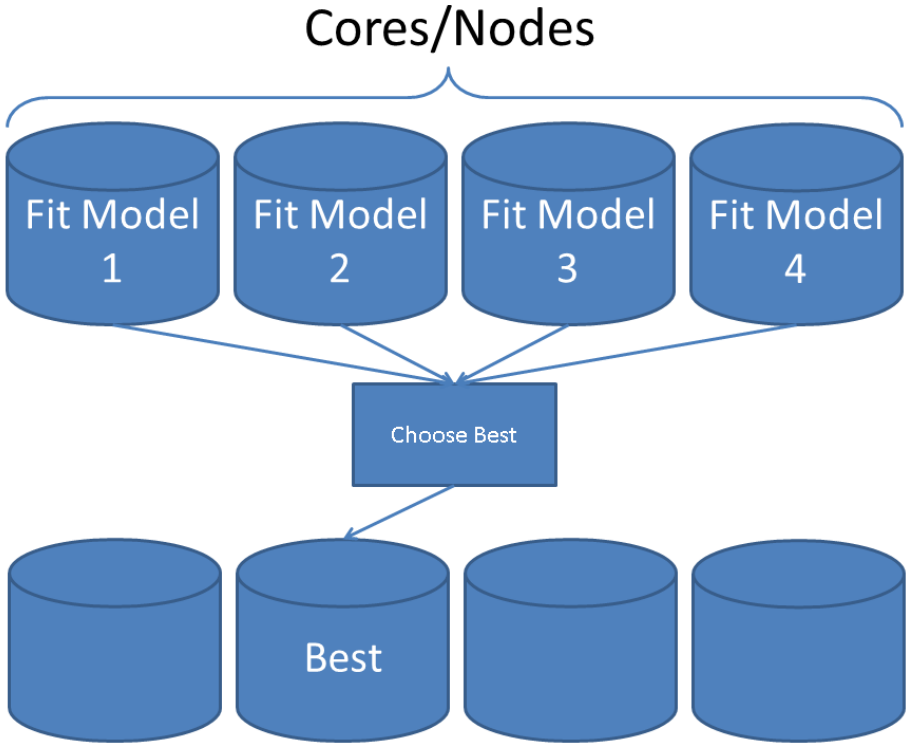


Figure 2: Task Parallelism

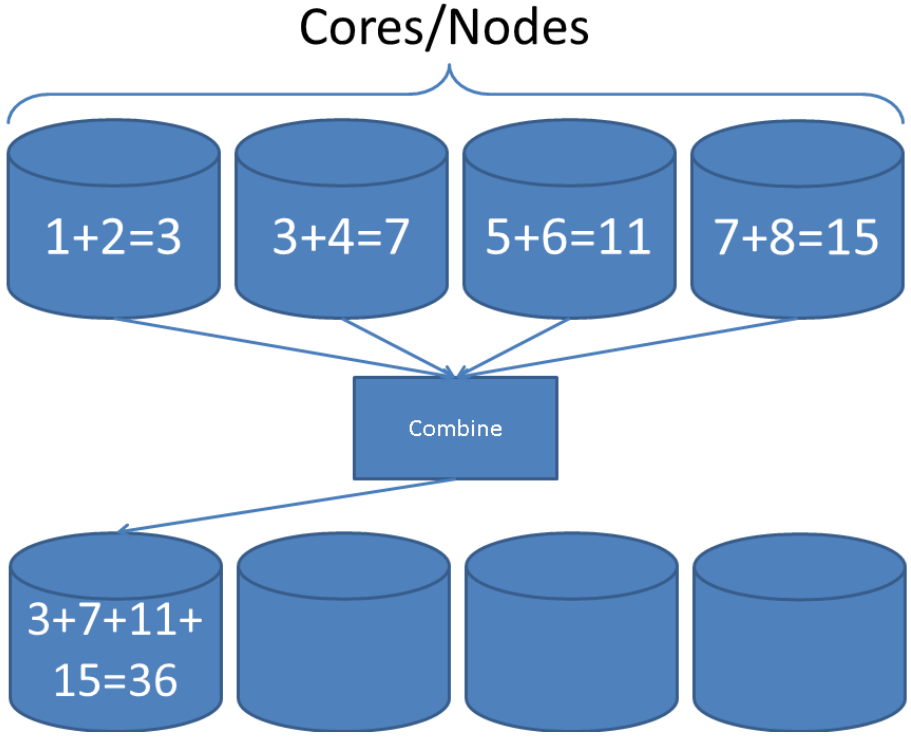


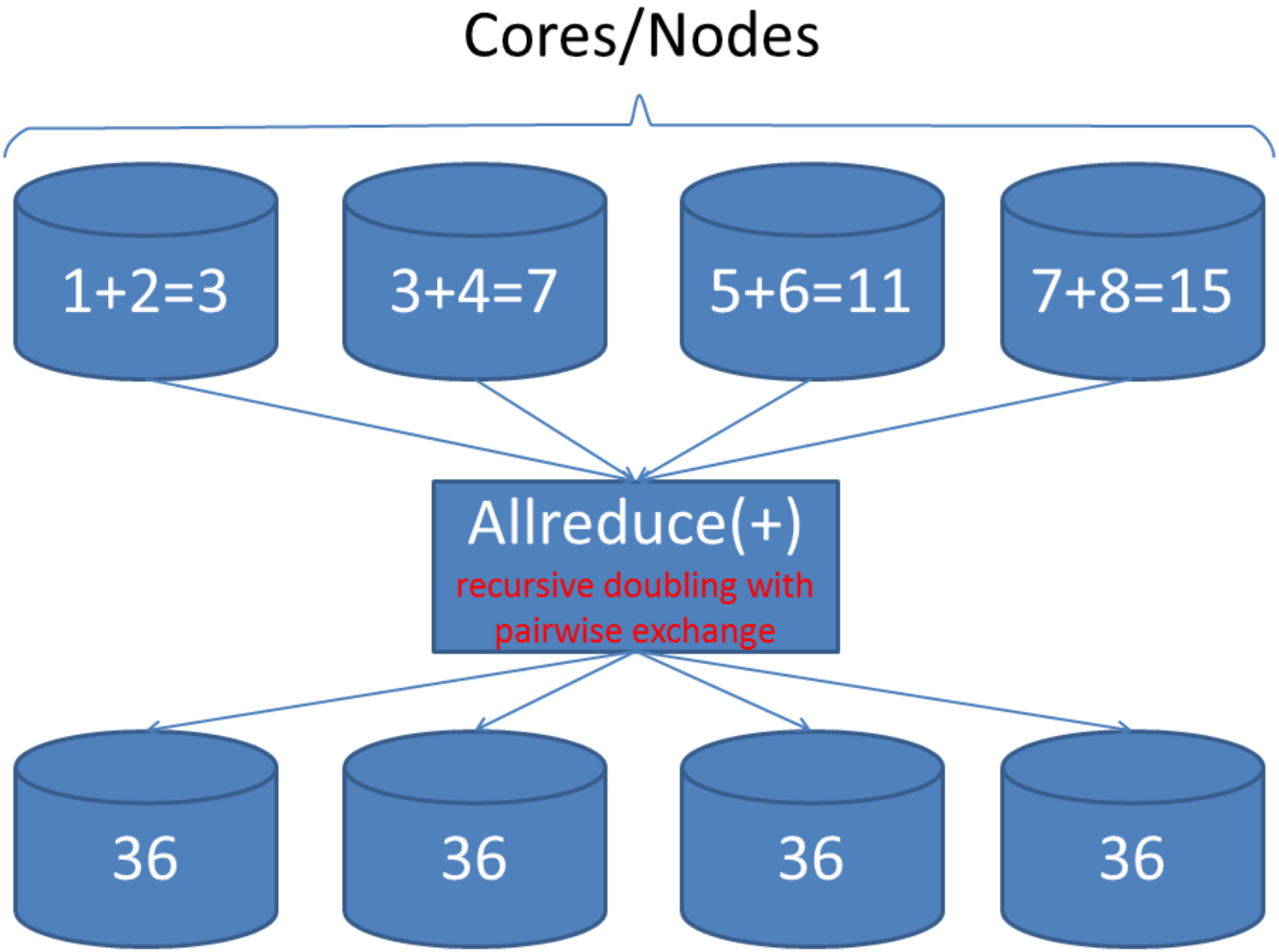
Figure 3: Data Parallelism

SPMD & MPI

SPMD in Wikipedia:
... In computing, SPMD (single program, multiple data) is a technique employed to achieve parallelism; ... *Tasks are split up and run simultaneously on multiple processors with different input in order to obtain results faster.* SPMD is the most common style of parallel programming ... *The current de facto standard is MPI* ...

MPI in Wikipedia:
Message Passing Interface (MPI) is a standardized and portable message-passing standard ... to function on a wide variety of parallel computing architectures... Language bindings R bindings of MPI include Rmpi and pbdMPI where Rmpi focuses on manager-workers parallelism while *pbdMPI focuses on SPMD parallelism*...

An SPMD Example



The Power of MPI Reduction Operations

$$X = \begin{pmatrix} X_0 \\ \vdots \\ X_7 \end{pmatrix} \text{ then } X^T X = \sum_{i=0}^{7} X_i^T X_i$$

```
Library(pbdMPI)
myrank = comm.rank()

## construct a matrix chunk
X = matrix(100*myrank + 1:8, nrow=4)

XtXloc = crossprod(X)

XtX = allreduce(XtXloc)

## everyone has XtX

finalize()
```

SPMD

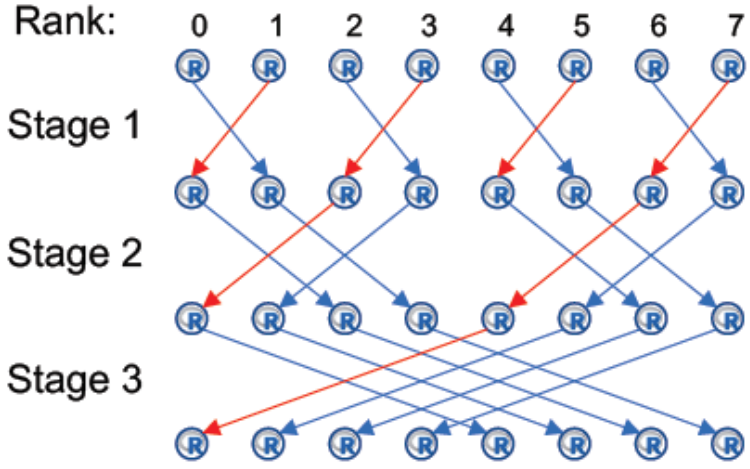
```
> myrank
[1] 0
> X
     [,1] [,2]
[1,]    1    5
[2,]    2    6
[3,]    3    7
[4,]    4    8
```

```
> myrank
[1] 7
> X
     [,1] [,2]
[1,]  701  705
[2,]  702  706
[3,]  703  707
[4,]  704  708
```

```
> XtX
         [,1]    [,2]
[1,] 5656240 5701360
[2,] 5701360 5746992
```

Recursive doubling with pairwise exchange

allreduce(), reduce(), allgather(), gather()



Communication stages when reduction is associative		
Processors	MapReduce shuffle	MPI allreduce
8	**7**	**3**
128	**127**	**8**
n	**n - 1**	**$\log_2 n$**

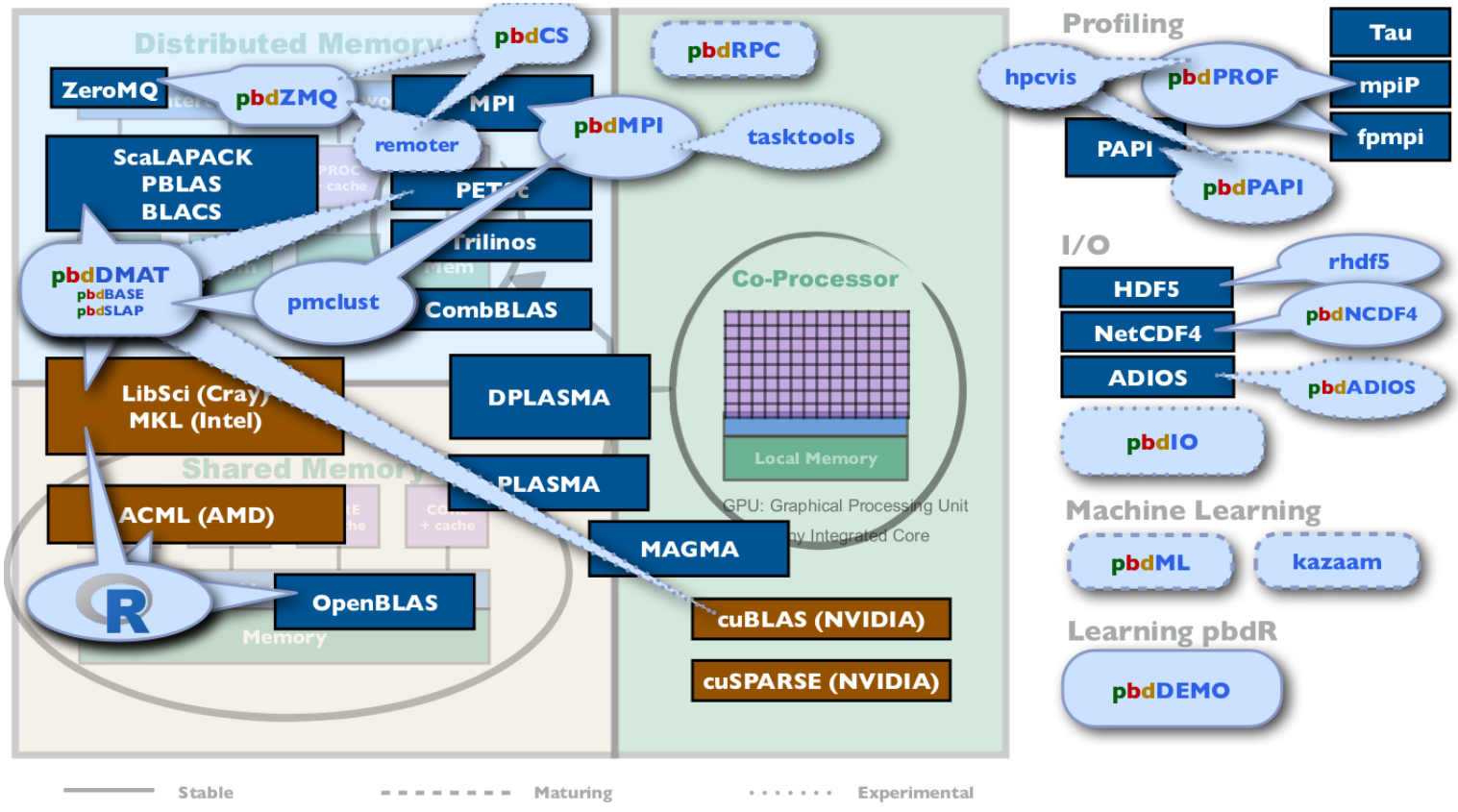
pbdR Thinking

Strive for *Productivity, Portability, Performance*

- Bridge high-performance computing with high-productivity of R language
- Keep syntax *identical* to native R , when possible
- Software reuse philosophy:
  - Don't reinvent the wheel when possible
  - Introduce HPC standards with R flavor
  - Use scalable HPC libraries with R convenience
- Simplify and use R intelligence where possible

pbdR v1.0-1 (Apr 2018) at `http://pbdr.org/releases/`

- MPI packages: pbdMPI, pbdSLAP, pbdBASE, pbdDMAT, kazaam, tasktools
- Statistical Applications: pbdML, pmclust, pbdDEMO
- Communication tools: pbdZMQ, remoter, pbdCS, pbdRPC
- Profilers: pbdPROF, pbdPAPI, hpcvis
- I/O packages: pbdIO, pbdNCDF4, pbdADIOS, hdfio



Dense Distributed Linear Algebra and Statistics

See vignettes of pbdBASE, pbdDMAT, and kazaam.

- pbdBASE and pbdDMAT provides block or block-cyclic distributed matrices, algebra, and operations.

- kazaam is designed for "tall but skinny" matrices where data are distributed in row blocks.

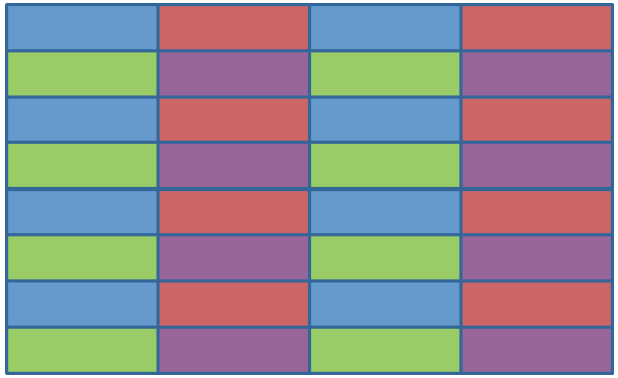


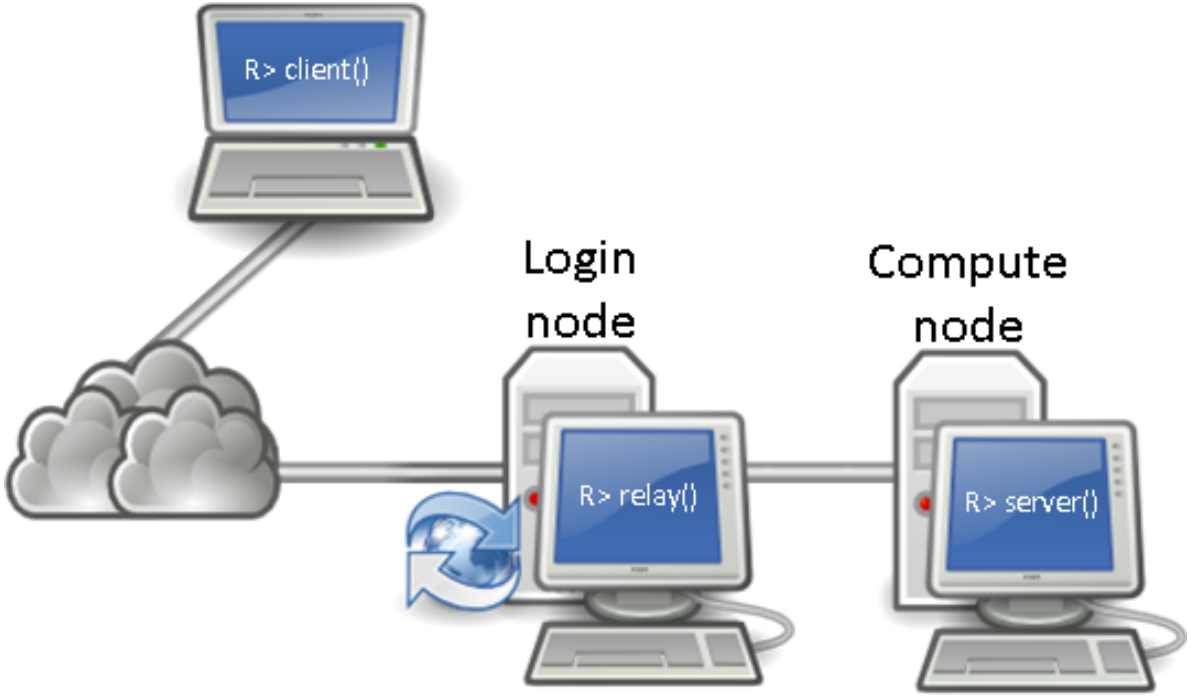
Figure 4: A 2D block cyclic matrix distributed in 4 cores (indicated by colors).

Figure 5: A tall but skinny matrix distributed in 4 cores (indicated by colors).

Client Server(s) in SPMD

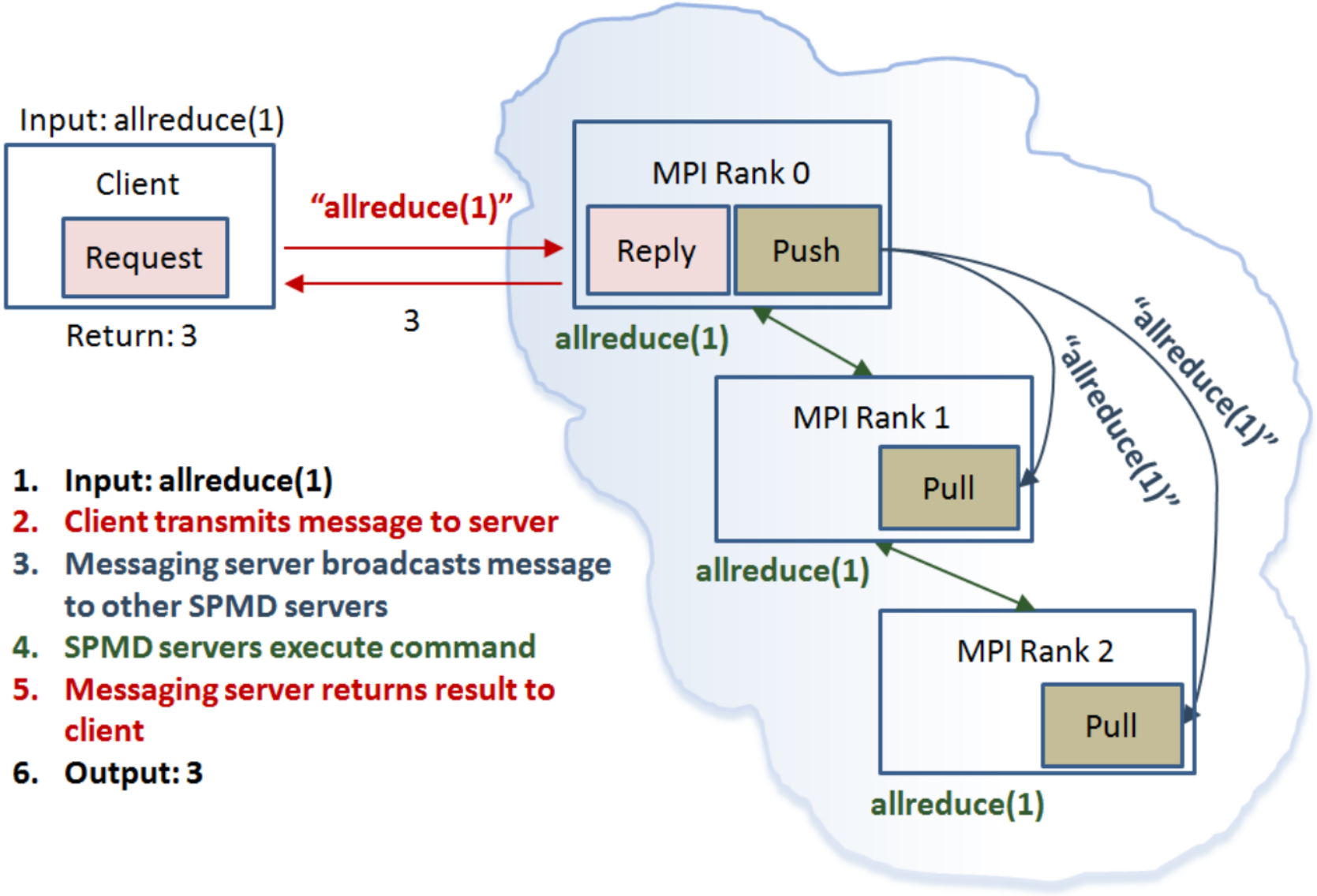
See demos at
https://github.com/snoweye/user2016.demo

- remoter + pbdZMQ: one client and one server
- pbdCS + remoter + pbdZMQ: one client and many MPI/SPMD servers, clusters, or HPC computing nodes
- pbdRPC: remote procedure calls to start the server(s)

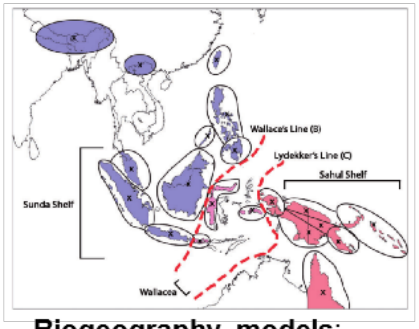


Basic Interaction with SPMD

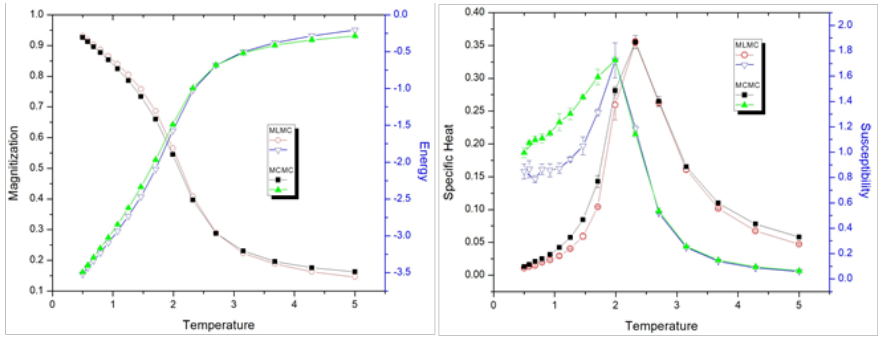
- Big data are obtained by and stays distributed on the servers
- The client sends only code to the server executing in SPMD



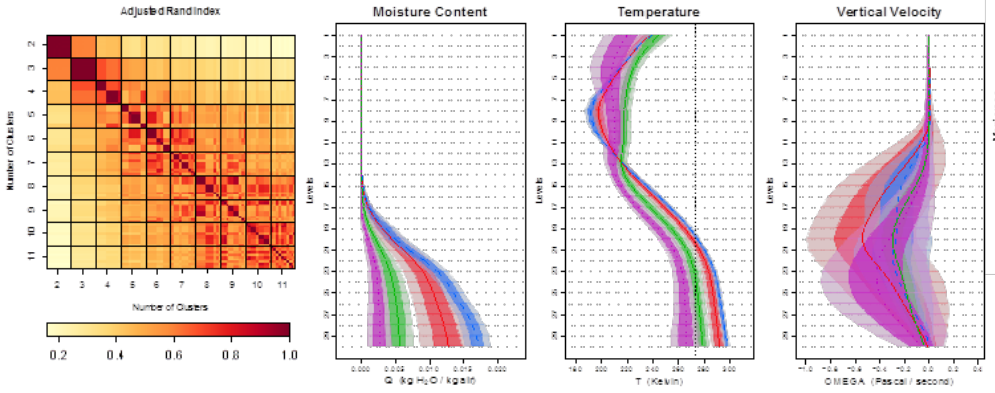
Diverse Statistical Applications of pbdR Analytics



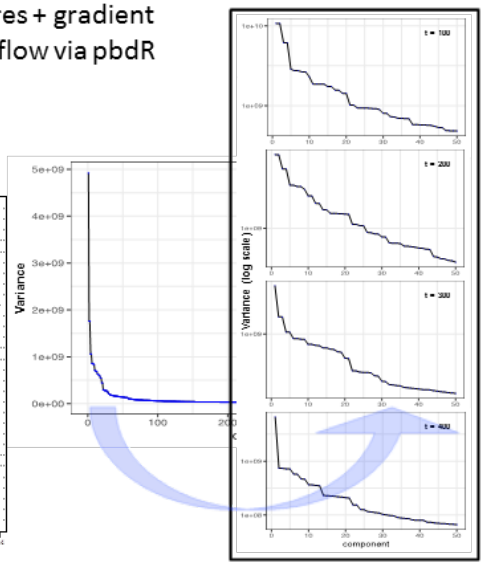
Biogeography models: distributed matrix exponentials (*UseR, 2014*)

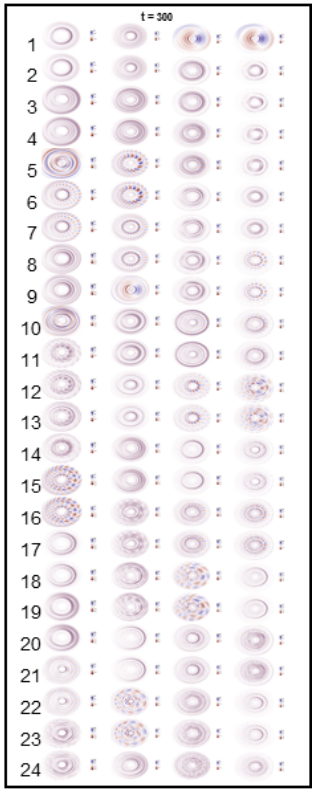


MCMC DFT: Deep neural network features + gradient boosting surrogate model. Parallel workflow via pbdR (*CoDA 2018*)



Climate extremes extraction and clustering with mixture models: scalable graphics, Rand index uncertainty, 3 TB data (*Technometrics, 2013*)





Fusion simulation: extracting coherent turbulence structures (*CoDA 2018*)

Statistical Algorithms and Methods

All algorithms and methods are in SPMD (i.e. parallel and distributed). Again, see pbdDEMO vignettes for more.

- Implemented:
  - matrix/vector operations, matrix decompositions, summary statistics, random number generation
  - pbdDMAT: linear model, logistic regression, SVM
  - kazaam: k-means, linear model, glm fitters (Gaussian, Logistic, Poisson)
  - pmclust: k-means, model-based clustering, EM/APECM algorithms
  - pbdML: random SVD/PCA, robust PCA, Fisher's linear discriminant
- Future plan:
  - pbdML: SVM, random forest, neural network, bagging, boosting, etc
  - *How about yours?!*

Computational Statistics Utilities

The tasktools is designed for task-based parallelism which is especially useful for multiple but independent simulations, MCMC, long searching for optimal study design.

- Include an lapply()-like interface
- More convenient than pbdMPI::pbdLapply() or Rmpi
- Automatically handles input-checkpointing:
  - Have thousands of ”jobs”
  - Run as many as you can in 2 hour run window
  - Keep running job until all tasks eventually complete
- Can be used as a workflow tool for external programs

Extensions to Data Science

User interfaces (R packages) developed by external groups that rely on pbdR (mainly pbdZMQ):

- IRkernel: Native R Kernel for the 'Jupyter Notebook' at `https://github.com/IRkernel/IRkernel`
- JuniperKernel: Kernel for 'Jupyter' at `https://github.com/JuniperKernel/JuniperKernel`

Summary

- Engage parallel math libraries at scale
- R language unchanged
- New distributed concepts
- New interactive SPMD parallel
- Broad analysis and statistical applications

Not included in this talk:

- New profiling capabilities
- Parallel and high performance I/O
- Comparisons with other frameworks:
  - A review paper: Thomas & Kumar (2018) “A Comparative Evaluation of Systems for Scalable Linear Algebra-based Analytics.”
  - From HPC wire, July 6, 2016: “OLCF Researchers Scale R to Tackle Big Science Data Sets” ... “**for ... interactive near-real-time analysis, the pbdR approach is much better** [than Apache Sparklike frameworks]. PCA of a 134 GB matrix: “hours on ... Apache Spark, ... less than a minute using R.”

Acknowledgement

This project, including software, documentation, talks, and tutorials, was supported in part by the following:

- Division of Mathematical Sciences, National Science Foundation, Award No. 1418195, 2014-2017.
- The National Institute for Mathematical and Biological Synthesis, under Award No. EF-0832858 and DBI-1300426, 2013-2014.
- The Division of Molecular and Cellular Biosciences, National Science Foundation Award MCB-1120370, 2013-2014.
- The Office of Cyberinfrastructure of the U.S. National Science Foundation under Award No. ARRA-NSF-OCI-0906324 for NICS-RDAV center, 2012-2013.
- U.S. Department of Energy Office of Science under Contract No. DE-AC05-00OR22725, 2011-2013.

We thank everyone who has submitted a bug report for the pbdR project. We also thank the members of the CRAN for their help and suggestions with pbdR packages, as well as their tireless efforts to develop and support R and its extensions.

Thank you!