Lecture 20 - MPI Part 3

DSE 512

Drew Schmidt
2022-04-07

From Last Time

- Homework 3
  - Originally due Saturday April 9
  - Now due Wednesday April 13
  - THERE WILL BE NO ADDITIONAL EXTENSIONS
- Did I forget something?
- Questions?

Today

- Data parallelism with MPI
    - SVD
    - GLM's
    - Some other approaches

Parallel SVD

Recall: Connection to Eigendecomposition

$$
\begin{aligned}
A^T A &= \left(U\Sigma V^T\right)^T \left(U\Sigma V^T\right) \\
&= V\Sigma U^T U\Sigma V^T \\
&= V\Sigma^2 V^T
\end{aligned}
$$

Recall: Computing the "Normal Equations" Matrix

Choose $b > 0$ and split A into b blocks of rows:

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_b \end{bmatrix}$$

Then

$$A^T A = \sum_{i=1}^{b} A_i^T A_i$$

Recall: Crossproduct-Based SVD Algorithms

Out-of-core

- Inputs
  - $A_{m \times n}$
  - Number of blocks b
- Procedure
  - Initialize $B_{n \times n} = 0$
  - For each $1 \leq i \leq b$
    - Read block of rows A_i
    - Compute $B = B + A_i^T A_i$
  - Factor $B = \Lambda \Delta \Lambda$

Parallel

- Inputs
  - $A_{m \times n}$
  - Number of cores c
- Procedure
  - Distribute matrix A among c workers into submatrices A_i
  - Compute $B_i = A_i^T A_i$
  - Compute $B = \sum_{i=1}^{c} B_c$
  - Factor $B = \Lambda \Delta \Lambda$

Parallel SVD

```
parsvd = function(A_local){
  B_local = crossprod(A_local)
  B = allreduce(B_local)
  e = eigen(B)

  sigma = sqrt(e$values)
  v = e$vectors

  u_local = sweep(v, STATS=1/sigma, MARGIN=2, FUN="*")
  u_local = A_local %*% u_local
  list(sigma=sigma,
    u_local = u_local,
    v = v
  )
}
```

Cute Trick

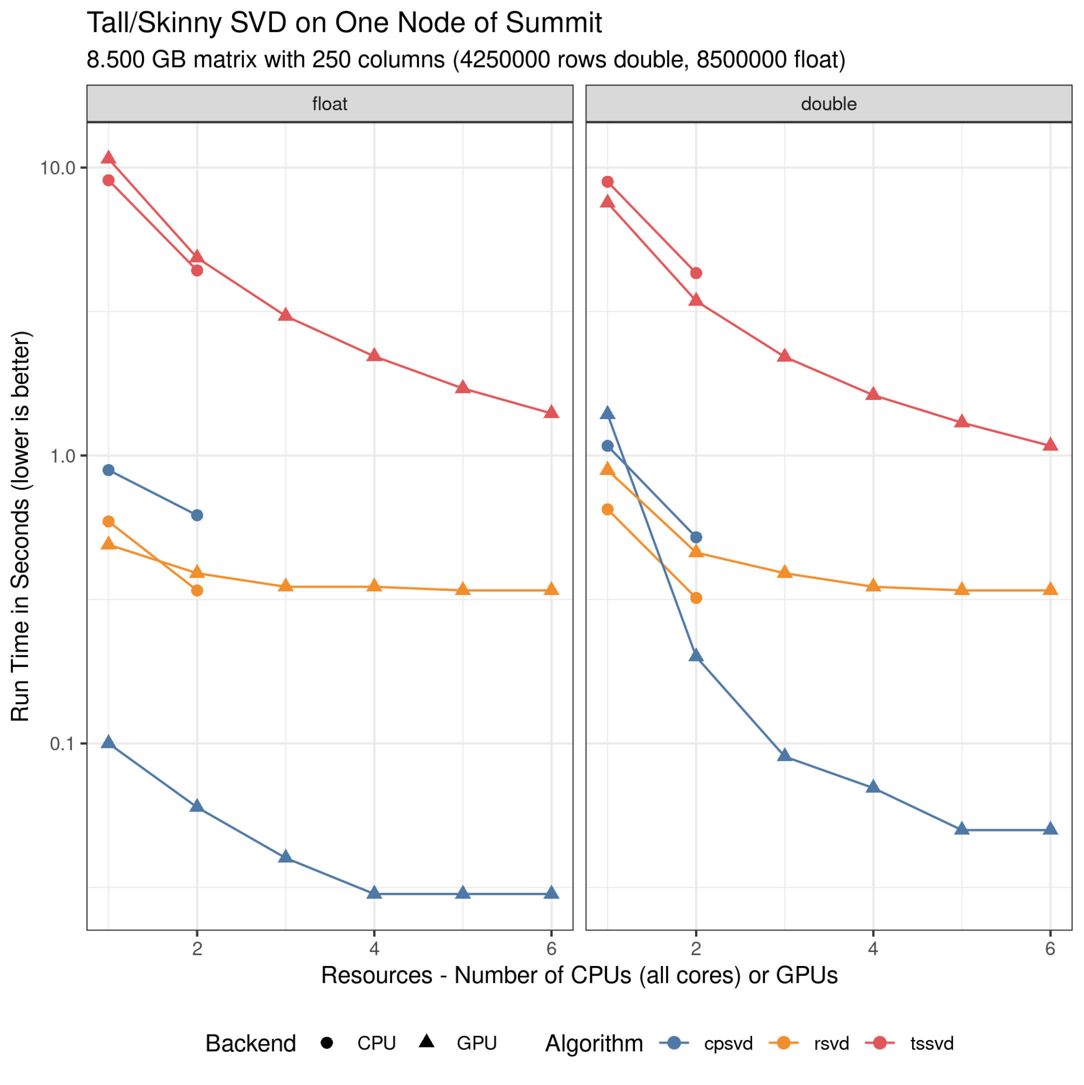
```
allreduce = identity
set.seed(1234)
A = matrix(rnorm(3*2), 3, 2)
parsvd(A)
```

```
## $sigma
## [1] 2.8602018 0.6868562
##
## $u_local
##            [,1]         [,2]
## [1,] -0.9182754 -0.359733536
## [2,]  0.1786546 -0.003617426
## [3,]  0.3533453 -0.933048068
##
## $v
##           [,1]       [,2]
## [1,] 0.5388308 -0.8424140
## [2,] 0.8424140  0.5388308
```

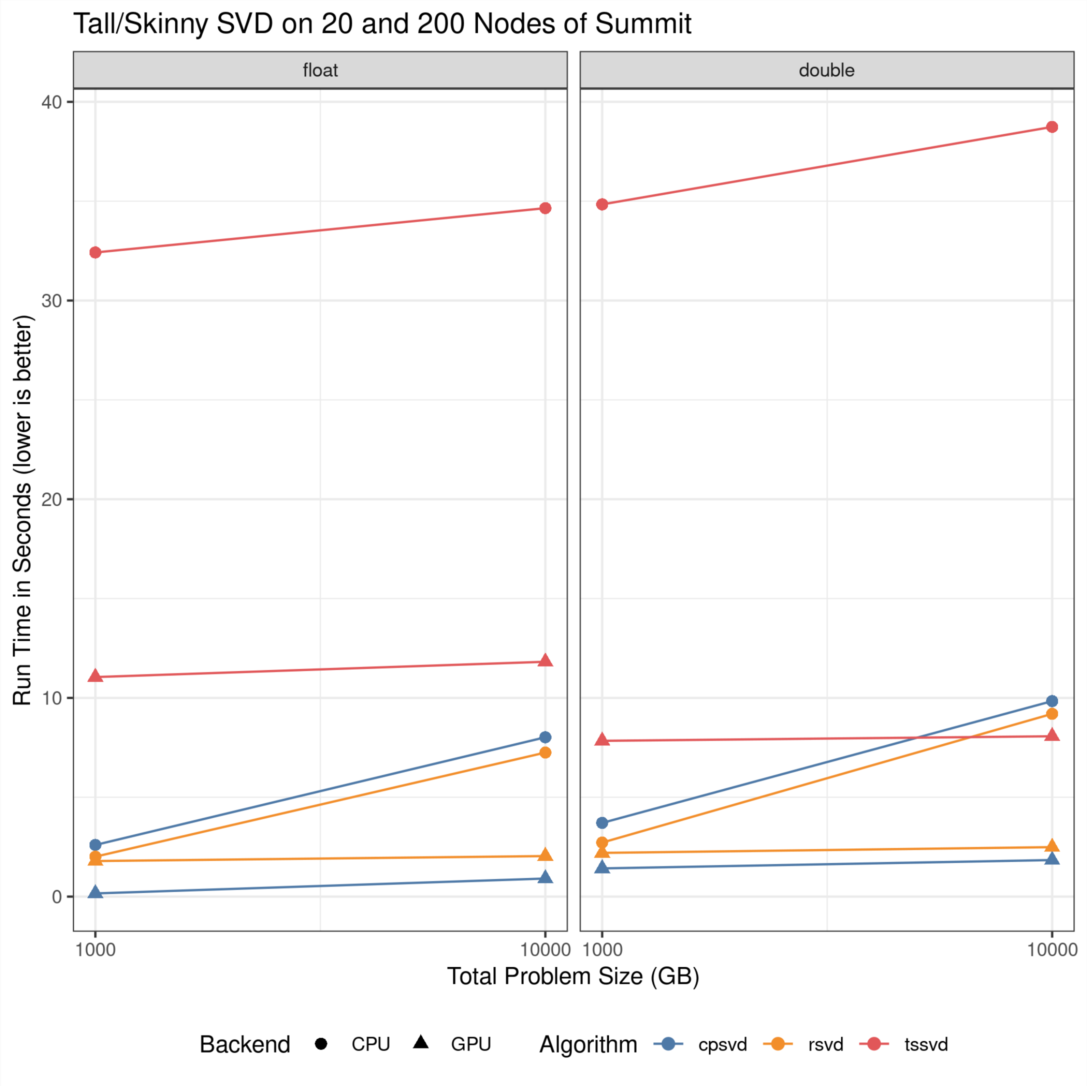
```
svd(A)
```

```
## $d
## [1] 2.8602018 0.6868562
##
## $u
##            [,1]         [,2]
## [1,] -0.9182754 -0.359733536
## [2,]  0.1786546 -0.003617426
## [3,]  0.3533453 -0.933048068
##
## $v
##           [,1]       [,2]
## [1,] 0.5388308 -0.8424140
## [2,] 0.8424140  0.5388308
```

Comparing SVD Implementations



Comparing SVD Implementations



More Information

See: Schmidt, D., 2020, November. A Survey of Singular Value Decomposition Methods for Distributed Tall/Skinny Data. In 2020 IEEE/ACM 11th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA) (pp. 27-34). IEEE.

Parallel Regression

Recall: Regression

- Normal equations
- QR
- SVD
- Solving the optimization problem

Recall: Solving the Regression Optimization Problem

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2m} \sum_{i=1}^{m} \left((X\beta)_i - y_i\right)^2$$

```
cost_gaussian = function(beta, x, y){
  m = nrow(x)
  (1/(2*m))*sum((x%*%beta - y)^2)
}

reg.fit = function(x, y, maxiter=100){
  control = list(maxit=maxiter)
  beta = numeric(ncol(x))
  optim(par=beta, fn=cost_gaussian, x=x, y=y, method="CG", control=control)
}
```

```
reg.fit(X, y)$par
```

Regression: Serial to Parallel

```r
cost_gaussian = function(beta, x, y){
  m = nrow(x)
  J = (1/(2m))sum((x%*%beta - y)^2)
  J
}
```

```r
cost_gaussian = function(beta, x, y){
  m = nrow(x)
  J_local = (1/(2m))sum((x%*%beta - y)^2)
  J = allreduce(J_local)
  J
}
```

- Global: `beta`
- Distributed: `x`, `y`

Logistic Regression

$$\min_{\theta \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} -y \log(g^{-1}(X\theta)) - (1-y)\log(1 - g^{-1}(X\theta))$$

```r
linkinv_logistic = binomial(logit)$linkinv

cost_logistic = function(theta, x, y)
{
  m = nrow(x)
  eta = x%*%theta
  h = linkinv_logistic(eta)
  (1/m)*sum((-y*log(h)) - ((1-y)*log(1-h)))
}

logistic.fit = function(x, y, maxiter=100)
{
  control = list(maxit=maxiter)
  theta = numeric(ncol(x))
  optim(par=theta, fn=cost_logistic, x=x, y=y, method="CG", control=control)
}
```

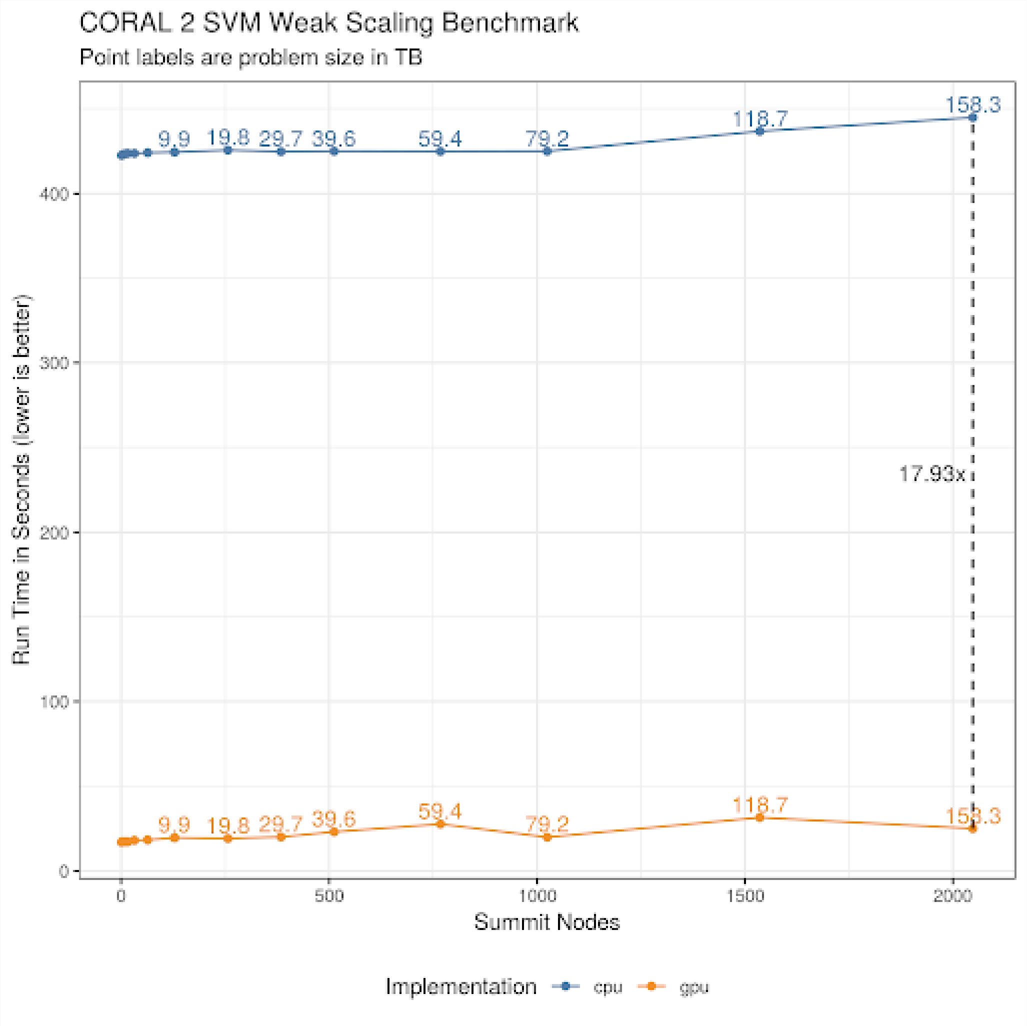
Logistic Regression: Serial to Parallel

```
cost_logistic = function(theta, x, y)
{
  m = nrow(x)
  eta = x%%theta
  h = linkinv_logistic(eta)
  J = (1/m)sum((-ylog(h)) - ((1-y)log(1-h
}
```

```
cost_logistic = function(theta, x, y)
{
  m = nrow(x)
  eta = x%%theta
  h = linkinv_logistic(eta)
  J_local = (1/m)sum((-ylog(h)) - ((1-y)lo
  J = allreduce_dbl(J_local)
  J
}
```

- Global: `theta`
- Distributed: `x`, `y`

Linear 2-Class SVM Benchmark



Other Approaches

Distributing a Matrix

$$x = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{bmatrix}_{9\times 9}$$

Distributing a Matrix: Blocking by Rows

$$x = \left[\begin{array}{ccccccccc}
x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\
x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\
x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\
\hline
x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\
x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\
x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\
\hline
x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\
x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\
x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99}
\end{array}\right]_{9\times 9}$$

$$\text{Processors} = \left[\begin{array}{cccc} 0 & 1 & 2 & 3 \end{array}\right]$$

Distributing a Matrix: Blocking by Columns

$$x = \left[\begin{array}{ccc|ccc|ccc} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{array}\right]_{9\times 9}$$

$$\text{Processors} = \begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}$$

Distributing a Matrix: Cycling by Rows

$$x = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ \hline x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ \hline x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ \hline x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ \hline x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ \hline x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ \hline x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ \hline x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{bmatrix}_{9 \times 9}$$

$$\text{Processors} = \begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}$$

Distributing a Matrix: Cycling by Columns

$$x = \left[\begin{array}{c|c|c|c|c|c|c|c|c} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{array}\right]_{9\times 9}$$

$$\text{Processors} = \begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}$$

Distributing a Matrix: Blocking by Rows AND Columns

$$x = \left[\begin{array}{ccccc|cccc} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ \hline x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{array}\right]_{9\times 9}$$

$$\text{Processors} = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

Distributing a Matrix: Cycling by Rows AND Columns

$$x = \left[\begin{array}{ccccc|cccc}
x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ \hline
x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ \hline
x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ \hline
x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ \hline
x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ \hline
x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ \hline
x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ \hline
x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ \hline
x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99}
\end{array}\right]_{9\times 9}$$

$$\text{Processors} = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

Distributing a Matrix: The 2-dimensional Block-Cyclic Layout

$$x = \left[\begin{array}{cc|cc|cc|cc|c}
x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\
x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ \hline
x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\
x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ \hline
x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\
x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ \hline
x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\
x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ \hline
x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99}
\end{array}\right]_{9\times 9}$$

$$\left[\begin{array}{cc|cc}
x_{11} & x_{12} & x_{17} & x_{18} \\
x_{21} & x_{22} & x_{27} & x_{28} \\ \hline
x_{51} & x_{52} & x_{57} & x_{58} \\
x_{61} & x_{62} & x_{67} & x_{68} \\ \hline
x_{91} & x_{92} & x_{97} & x_{98}
\end{array}\right]_{5\times 4}
\left[\begin{array}{cc|c}
x_{13} & x_{14} & x_{19} \\
x_{23} & x_{24} & x_{29} \\ \hline
x_{53} & x_{54} & x_{59} \\
x_{63} & x_{64} & x_{69} \\ \hline
x_{93} & x_{94} & x_{99}
\end{array}\right]_{5\times 3}
\left[\begin{array}{cc}
x_{15} & x_{16} \\
x_{25} & x_{26} \\ \hline
x_{55} & x_{56} \\
x_{65} & x_{66} \\ \hline
x_{95} & x_{96}
\end{array}\right]_{5\times 2}$$

$$\left[\begin{array}{cc|cc}
x_{31} & x_{32} & x_{37} & x_{38} \\
x_{41} & x_{42} & x_{47} & x_{48} \\ \hline
x_{71} & x_{72} & x_{77} & x_{78} \\
x_{81} & x_{82} & x_{87} & x_{88}
\end{array}\right]_{4\times 4}
\left[\begin{array}{cc|c}
x_{33} & x_{34} & x_{39} \\
x_{43} & x_{44} & x_{49} \\ \hline
x_{73} & x_{74} & x_{79} \\
x_{83} & x_{84} & x_{89}
\end{array}\right]_{4\times 3}
\left[\begin{array}{cc}
x_{35} & x_{36} \\
x_{45} & x_{46} \\ \hline
x_{75} & x_{76} \\
x_{85} & x_{86}
\end{array}\right]_{4\times 2}$$

local storage

$$\text{Processors} = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix} = \begin{bmatrix} (0,0) & (0,1) & (0,2) \\ (1,0) & (1,1) & (1,2) \end{bmatrix}$$

ScaLAPACK

- The scalable LAPACK
- Uses 2-d block-cyclic layout
- Created in the 90's
- Several attemps to replace it

Bindings

- Julia
  - ScaLAPACK.jl
- Python
  - scalapy
- R
  - pbdDMAT
  - fmlr

SVD with fmlr

```r
suppressMessages(library(fmlr))

g = grid()
x = mpimat(g, 5, 5, 1, 1)
x$fill_rnorm()
x$info()

s = cpuvec()
linalg_svd(x, s)
if (g$rank0()){
  s
}
```

```
mpirun -np 4 Rscript x.r
```

```
# mpimat 5x5 with 1x1 blocking on 2x2 grid type=d
3.1585 2.6992 1.2946 0.9501 0.5486
```

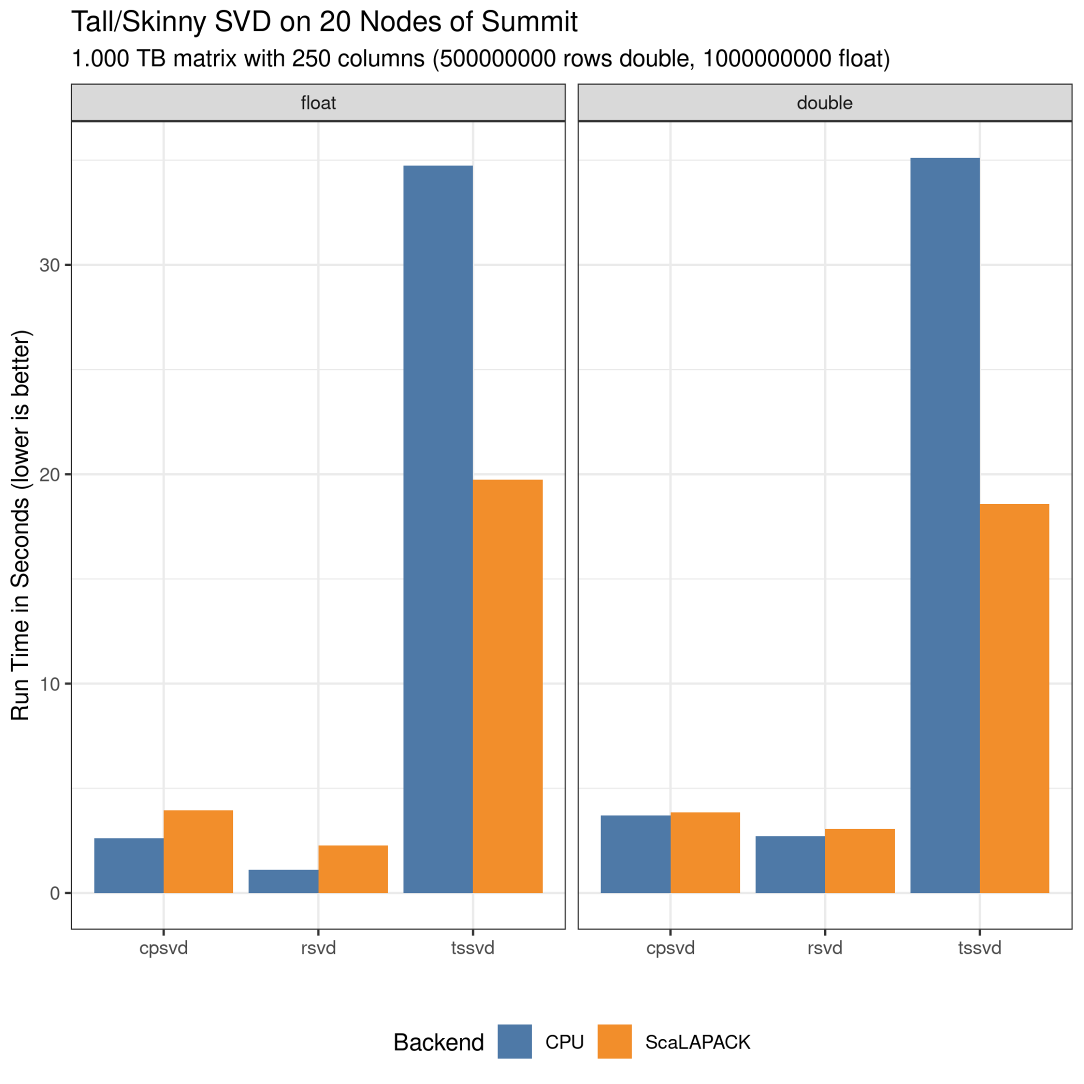
SVD with fmlr

```r
suppressMessages(library(fmlr))

g = grid()
x = mpimat(g, 5, 5, 1, 1)
x$fill_rnorm()
x$info()

s = cpuvec()
linalg_rsvd(1234, 1, 2, x, s)
if (g$rank0()){
  s
}
```

```
mpirun -np 4 Rscript x.r
```

```
# mpimat 5x5 with 1x1 blocking on 2x2 grid type=d
3.1162
```

Comparing SVD Implementations



Wrapup

Wrapup

- SPMD makes short work of data parallelism problems.
- The `allreduce()` collective is *very* powerful.
- Distributing by row/col is easy, but not always appropriate.
- Next time: the MapReduce algorithm

Questions?