GPU accelerated Bayesian mixture models for FCM analysis

Cliburn Chan, Jacob Frelinger, Adam Richards, Lin Lin, Ioanna Manolopoulou, Andrew Cron, Quanli Wang, Janet Ottinger, Kent Weinhold, Marc Suchard, Mike West
Duke University
Outline

• Research context
• Modeling FCM data
• GPU programming
• Python extensions
• Analysis of FlowCAP data
Polychromatic Flow

Basic subsets

Maturational subsets

Functional subsets (CD4+CD8-)

CD107+ IFNγ IL2 TNFα
Data characteristics

- Fairly high-dimensional (~10 colors)
- Large data sets (~$10^6$ events)
- Interest in rare events (~0.01%)
Probability distribution
Build from many simple distributions
Standard model

\[ f(x) = \sum_{j=1}^{T} \pi_j N(x | \mu_j, \Sigma_j). \]

\[ \eta_j \sim Gamma \left( \frac{a}{2}, \frac{a}{2} \right) \text{ where } E(\eta_j) = 1 \]

\[ \Sigma_j \sim IW(\nu + 2, \nu \eta_j \Phi_0) \text{ where } E(\Sigma_j) = \eta_j \Phi_0 \]

\[ \mu_j \sim N(m_0, \gamma \Sigma_j) \]

\[ \alpha \sim Gamma(e, f) \]

\[ v_j \sim Beta(1, \alpha) \]

\[ \pi_i = v_j \prod_{k=1}^{j-1} (1 - v_k) \]
Issues

- Fitting GMMs to large data is painfully slow
- Sub-sampling may miss rare event clusters
- Biased sampling can be helpful
The rise of GPUs
Why are GPUs faster?

- 100s of processing cores per chip
- Performing same computations on different data
- Single program, multiple data (SPMD) paradigm
Basic GPU workflow

- Copy data CPU => GPU global memory
- Transfer data global => shared memory
- Perform computation on GPU
- Write back to GPU global memory
- Copy data GPU global memory => CPU
GPU threads

• Where did the time go? (Profiler)
  
  – Must Compute NxJ normal densities (MCMC and EM)
    • Prime candidate for massive parallelization and data sharing
  
  – Assign Data Categories via N independent scan-reductions (MCMC)
    • Still Parallelizable, but still some unavoidable serial computations
  
  – Calculate covariance estimates for each component (Bayesian EM)
GPU threads

• Where did the time go? (Profiler)
  – Must Compute NxJ normal densities (MCMC and EM)
    • Prime candidate for massive parallelization and data sharing
  – Assign Data Categories via N independent scan-reductions (MCMC)
    • Still Parallelizable, but still some unavoidable serial computations
  – Calculate covariance estimates for each component (Bayesian EM)
Using registers and shared memory
Optimal reads from GPU global memory

Global Memory Transactions can take up to 600 clock cycles!
Optimal reads from GPU global memory

Coalesced Memory Access: 1 transaction.

Global Memory Transactions can take up to 600 clock cycles!
Optimal reads from GPU global memory

Coalesced Memory Access: 1 transaction.

Global Memory Transactions can take up to 600 clock cycles!

Non-sequential Memory Access: 16 transactions.
Is it worth the effort?

• Estimated times to compute 10,000 MCMC iterations
• 256 components, 14 dimensions
• Desktop: dual 4-core CPUs and 3 240-core GTX285 GPUs
• MacBook Pro: 32-core GT120 GPU
Is it worth the effort?

- Estimated times to compute 10,000 MCMC iterations
- 256 components, 14 dimensions
- Desktop: dual 4-core CPUs and 3 240-core GTX285 GPUs
- MacBook Pro: 32-core GT120 GPU

<table>
<thead>
<tr>
<th>N</th>
<th>Multi-CPU</th>
<th>1 GPU</th>
<th>3 GPUs</th>
<th>Base Time</th>
<th>GPU</th>
<th>Base Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^2$</td>
<td>1x</td>
<td>2.5x</td>
<td>2.4x</td>
<td>5 mins</td>
<td>2.5x</td>
<td>8 mins</td>
</tr>
<tr>
<td>$10^3$</td>
<td>1x</td>
<td>14x</td>
<td>15x</td>
<td>30 mins</td>
<td>8x</td>
<td>50 mins</td>
</tr>
<tr>
<td>$10^4$</td>
<td>2x</td>
<td>60x</td>
<td>78x</td>
<td>5 hrs</td>
<td>15x</td>
<td>8 hrs</td>
</tr>
<tr>
<td>$10^5$</td>
<td>5x</td>
<td>93x</td>
<td>146x</td>
<td>2 days</td>
<td>16x</td>
<td>3 days</td>
</tr>
<tr>
<td>$10^6$</td>
<td>5x</td>
<td>100x</td>
<td>160x</td>
<td>22 days</td>
<td>25x</td>
<td>32 days</td>
</tr>
</tbody>
</table>
Is it worth the effort?

- Estimated times to compute 10,000 MCMC iterations
- 256 components, 14 dimensions
- Desktop: dual 4-core CPUs and 3 240-core GTX285 GPUs
- MacBook Pro: 32-core GT120 GPU

<table>
<thead>
<tr>
<th>Desktop</th>
<th>Multi-CPU</th>
<th>1 GPU</th>
<th>3 GPUs</th>
<th>Base Time</th>
<th>Mac Laptop</th>
<th>GPU</th>
<th>Base Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^2$</td>
<td>1x</td>
<td>2.5x</td>
<td>2.4x</td>
<td>5 mins</td>
<td>2.5x</td>
<td>8 min</td>
<td></td>
</tr>
<tr>
<td>$10^3$</td>
<td>1x</td>
<td>14x</td>
<td>15x</td>
<td>30 mins</td>
<td>8x</td>
<td>50 min</td>
<td></td>
</tr>
<tr>
<td>$10^4$</td>
<td>2x</td>
<td>60x</td>
<td>78x</td>
<td>5 hrs</td>
<td>15x</td>
<td>8 hrs</td>
<td></td>
</tr>
<tr>
<td>$10^5$</td>
<td>5x</td>
<td>93x</td>
<td>146x</td>
<td>2 days</td>
<td>16x</td>
<td>3 days</td>
<td></td>
</tr>
<tr>
<td>$10^6$</td>
<td>5x</td>
<td>100x</td>
<td>160x</td>
<td>22 days</td>
<td>25x</td>
<td>32 days</td>
<td></td>
</tr>
</tbody>
</table>

Bottom Line: 22 days is reduced to 3 hours!
CPU versus GPU
The very alpha \textit{fcm} library

- GPU code wrapped for use in R, Matlab and Python
- \textit{fcm} = Python wrapped library
- Combines speed of GPU computing with ease-of-use of dynamic language
- Access to Python libraries for science, numerics, graphics, databases, XML, ...
Using *fcm* for GvHD data set

```python
import fcm
import fcm.statistics as stats
import pylab

# load data
x = fcm.loadFCS('GvHD/001.fcs')

# specify model
model = stats.DPMixtureModel(x, nclusts=16, iter=1000, burnin=0, last=5)

# fit model
model.fit()

# get classification labels using modes to merge components
labels = model.get_results().make_modal().classify(x)

# save labels in text file, one per line
pylab.savetxt('GvHD/001.txt', labels, delimiter='\n')
```
Some functionality

- compensation, transforms, projections
- visualizations
  - overlay histograms, density plots, contours, 3D surface, 3D spin plots
- limited interactive gating
- posterior summaries
  - modes, marker “usefulness”
- sample from prior and posterior distributions
fcm gallery

Tuesday, September 21, 2010
fcm in pipeline GUI
FlowCAP analysis

• Challenge 1 - 64 components, no exclusions, uninformative priors

• Challenge 2 - 16 components, no exclusions, uninformative priors

• Challenge 3 - 16 components, exclude scatter channels and events on axes, uninformative priors

• Challenge 4 - 16 components, exclude scatter channels and events on axes, informative priors from given sample labels
Issues

- Flow lab initially interested in helping us interpret data ... unfortunately, had lots of problems reading data with FlowJo and became frustrated instead
HSCT scatters

HSCT

Data Gaps

DLBCL

CFSE

CD45.1+

CD45.1-CD45.2-

CD45.2+

Viable Cells

PI Neg Cells

CD45.2+

CD45.1+

Ly65_MAC-1+

CFSE

Tuesday, September 21, 2010
Lessons

• Need more complete suite of tests for code - in one case, we had the same label for every single event due to a bug

• Sometimes, code can be too optimized - due to GPU code optimizations, need maximal components in multiples of 16

• Alternatives not evaluated yet
  • different choices of prior settings
  • different merge strategies
  • usefulness of supervised learning
Acknowledgements

- The fabulous people in Duke statistics, computational biology and surgical sciences who have worked with, and spent countless hours trying to educate me, especially the groups led by Mike West, Tom Kepler and Kent Weinhold

- Generous funding from the NIH that has supported various phases of this work (P30-AI064518-05, N01-AI 5000192662005, UL1-RR024128-04, RC1AI086032-01)