

GPU accelerated Bayesian mixture models for FCM analysis

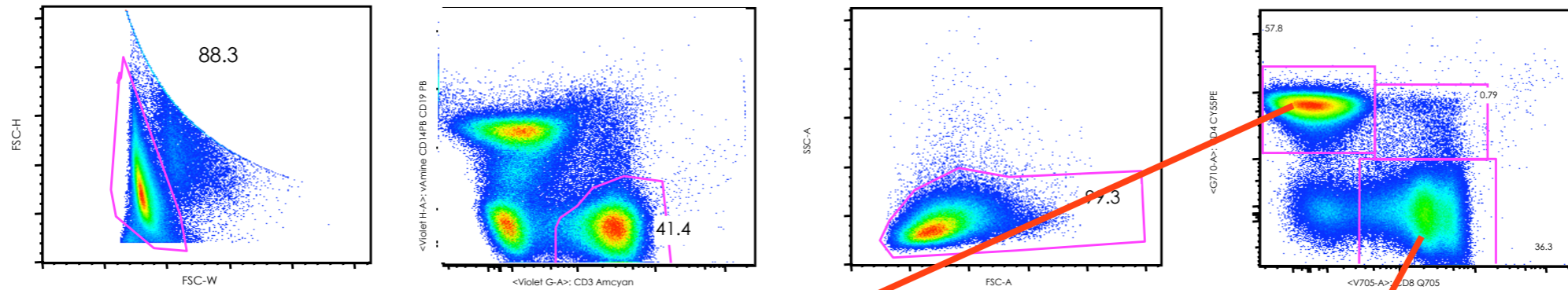
Cliburn Chan, Jacob Frelinger, Adam Richards, Lin Lin, Ioanna Manolopoulou, Andrew Cron, Quanli Wang, Janet Ottinger, Kent Weinhold, Marc Suchard, Mike West
Duke University

Outline

- Research context
- Modeling FCM data
- GPU programming
- Python extensions
- Analysis of FlowCAP data

Polychromatic Flow

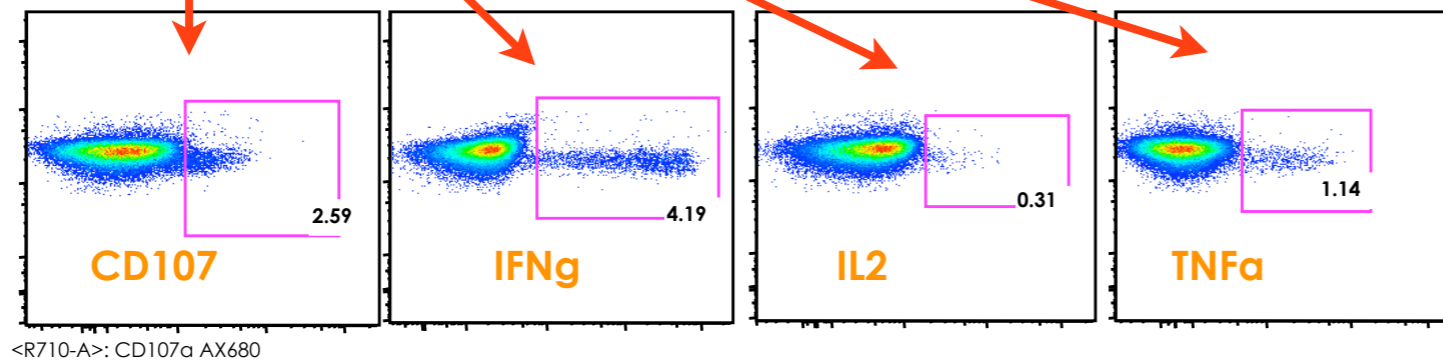
Basic subsets



Maturation subsets



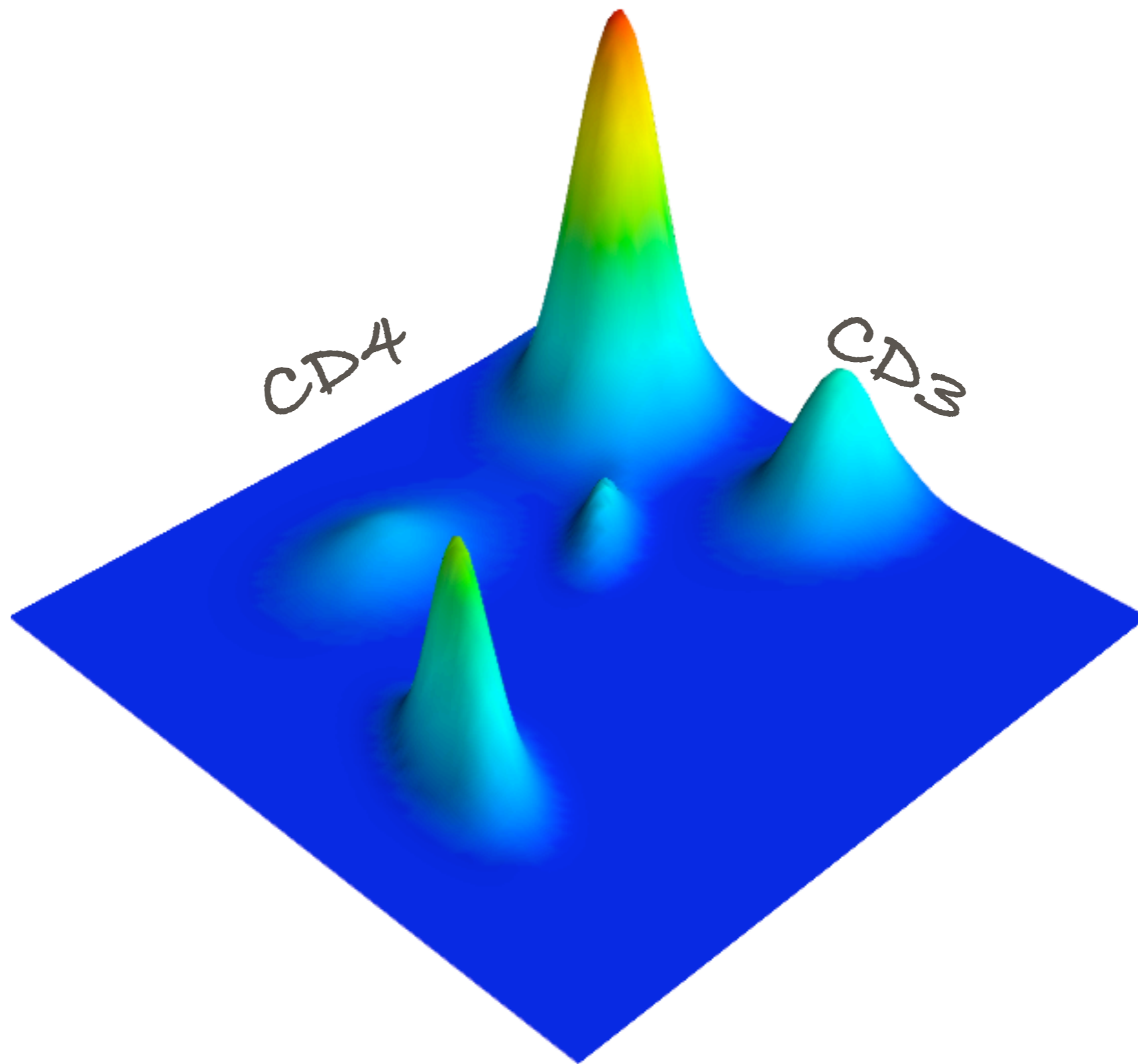
Functional subsets (CD4+CD8-)



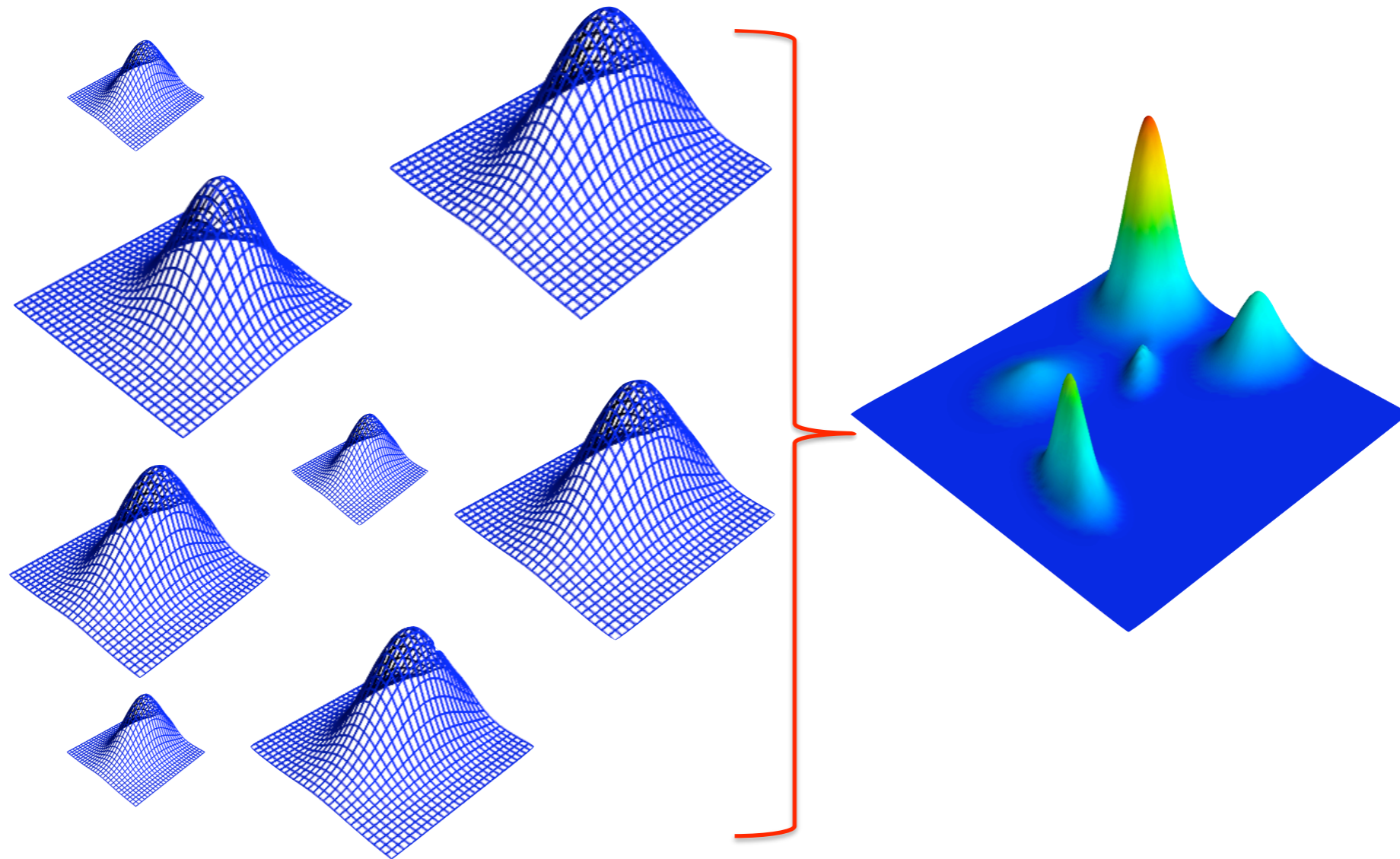
<R710-A>: CD107a AX680

Data characteristics

- Fairly high-dimensional (~ 10 colors)
- Large data sets ($\sim 10^6$ events)
- Interest in rare events ($\sim 0.01\%$)



Probability distribution



Build from many simple distributions

Standard model

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

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

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

$$\mu_j \sim N(m_0, \gamma \Sigma_j)$$

$$\alpha \sim \text{Gamma}(e, f)$$

$$v_j \sim \text{Beta}(1, \alpha)$$

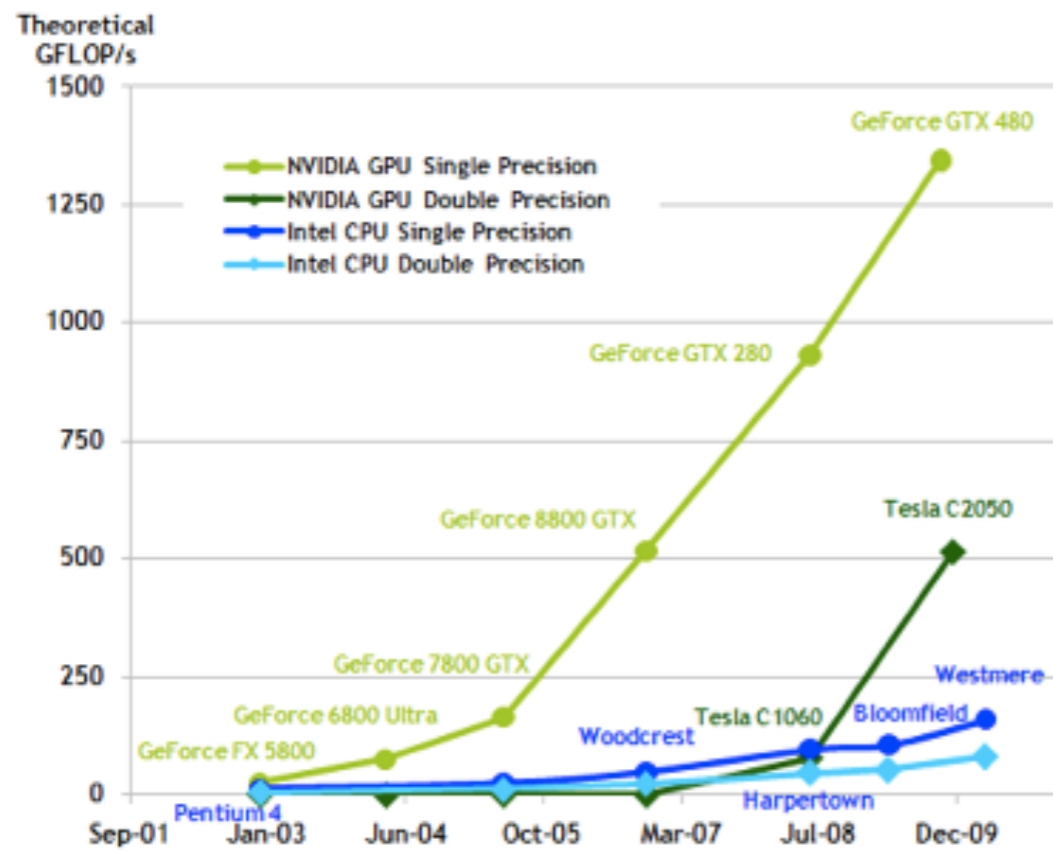
$$\pi_i = v_j \prod_{k=1}^{j-1} (1 - v_k)$$

Issues

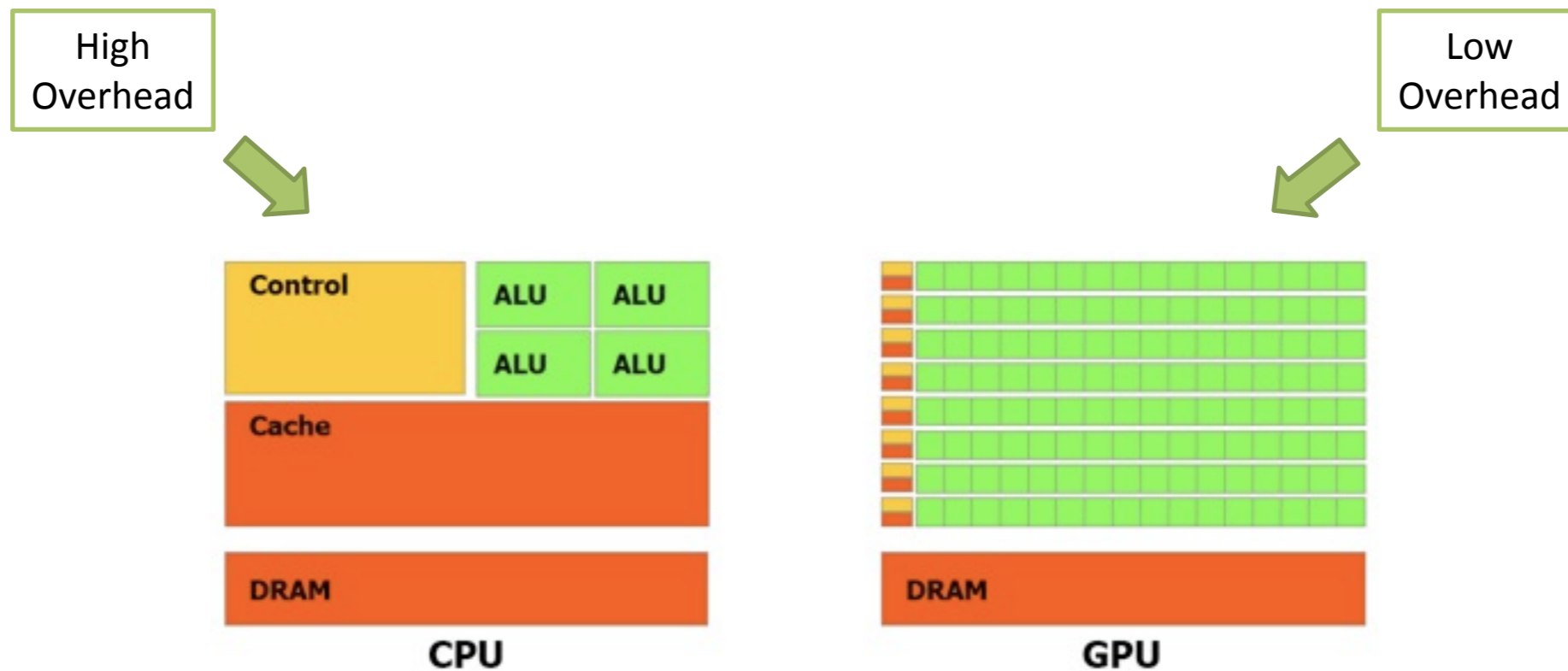
- Fitting GMMs to large data is painfully slow
- Sub-sampling may miss rare event clusters
- Biased sampling can be helpful



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

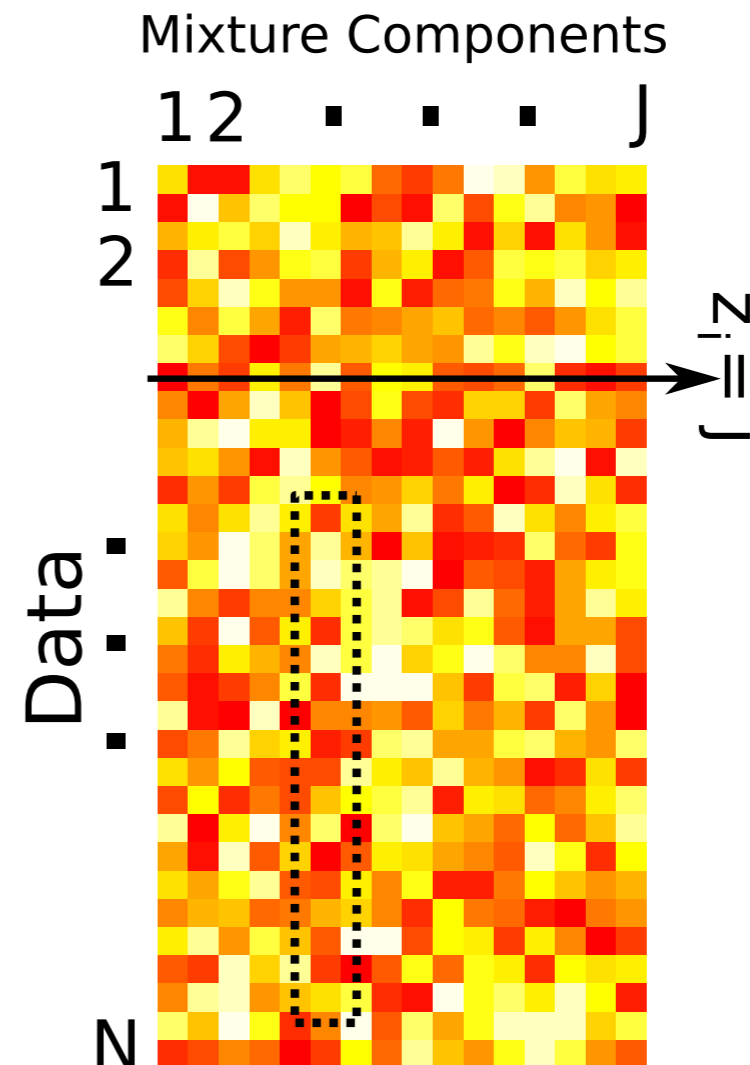
- Copy data CPU => GPU global memory
- Transfer data global => shared memory
- Perform computation on GPU
- Write back to GPU global memory
- Copy data GPU global memory => CPU

GPU threads

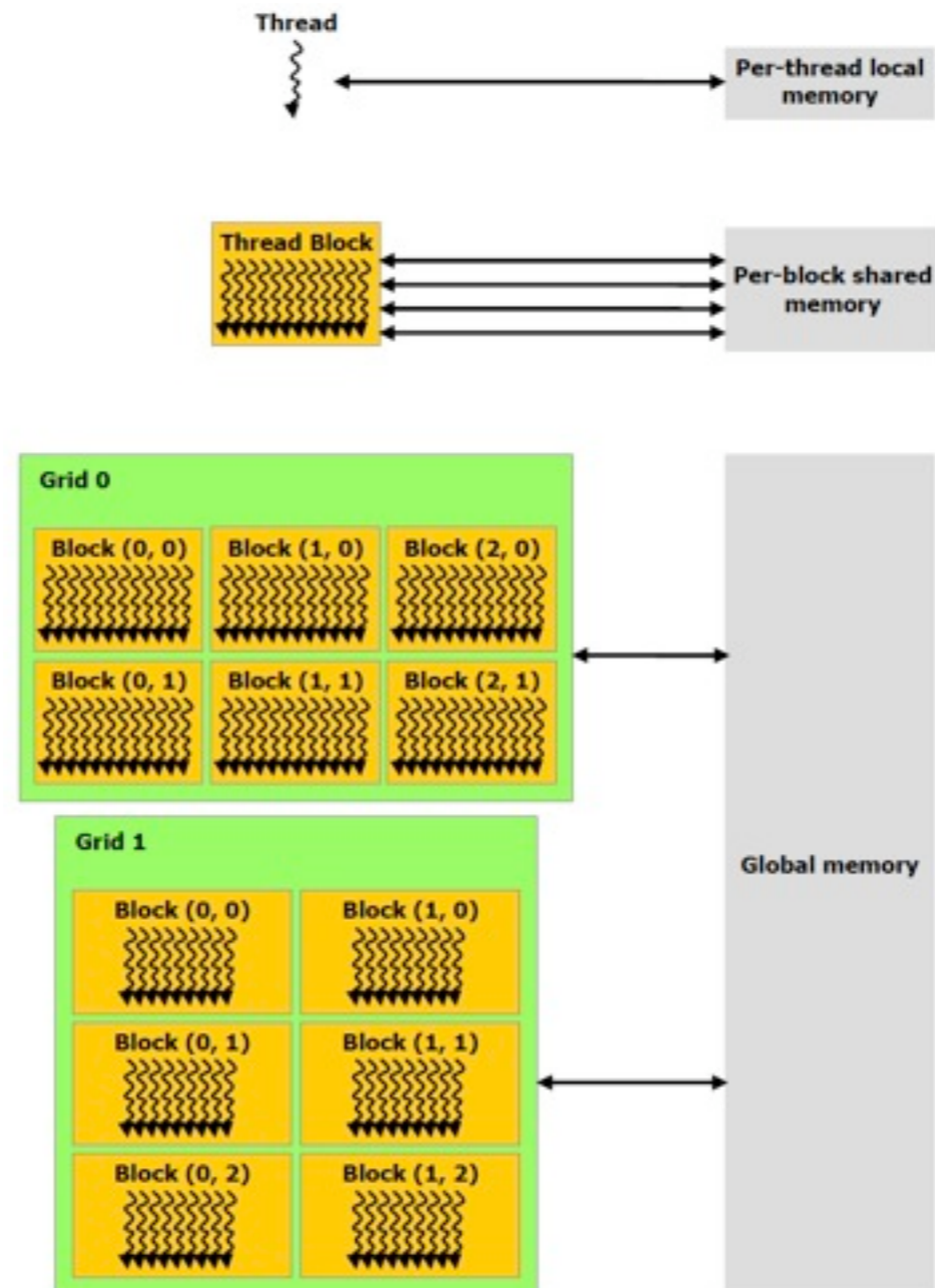
- Where did the time go? (Profiler)
 - Must Compute $N \times J$ normal densities (MCMC and EM)
 - Prime candidate for massive parallelization and data sharing
 - Assign Data Categories via N independent scan-reductions (MCMC)
 - Still Parallelizable, but still some unavoidable serial computations
 -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

GPU threads

- Where did the time go? (Profiler)
 - Must Compute $N \times J$ 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!

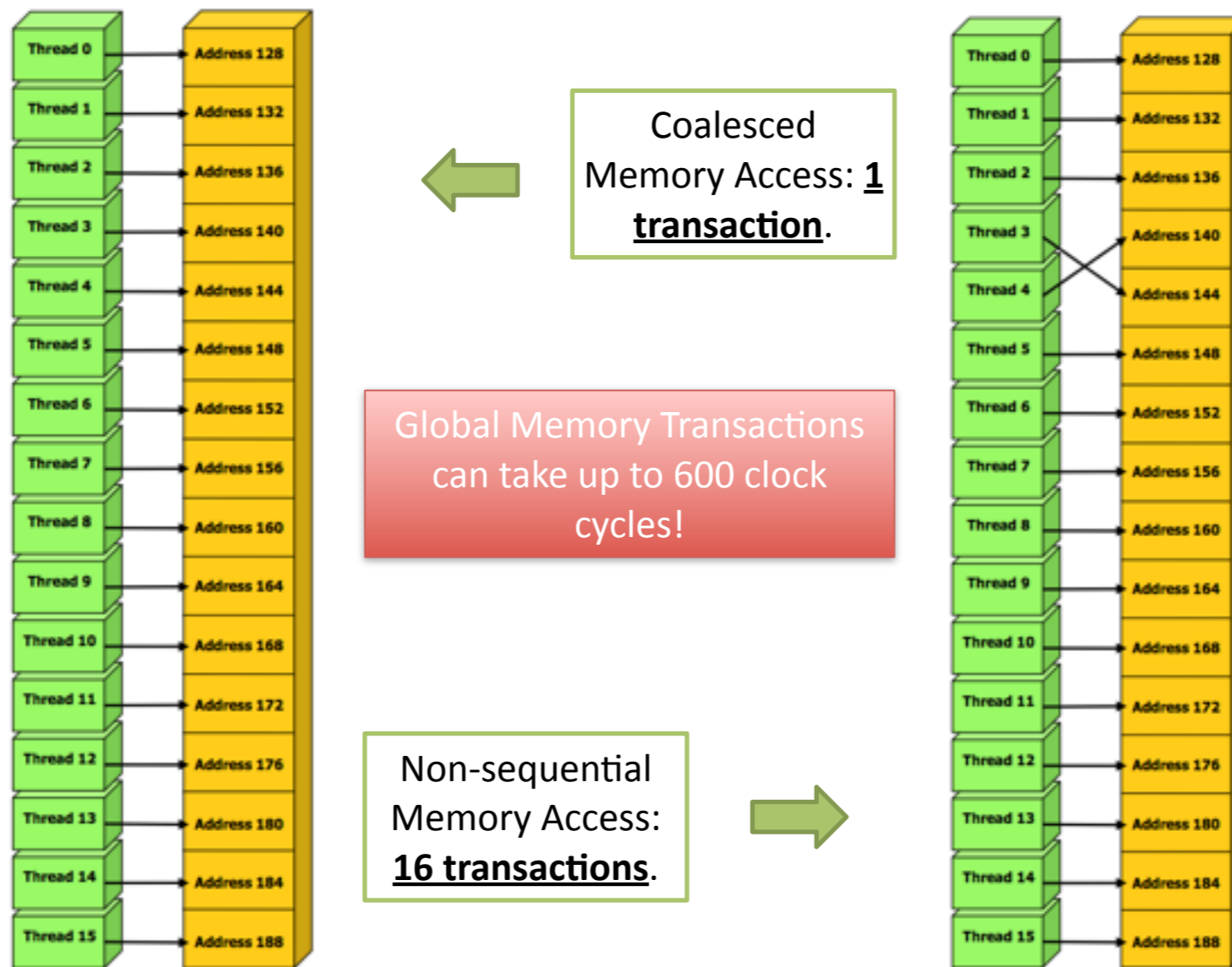
Optimal reads from GPU global memory



Coalesced
Memory Access: 1
transaction.

Global Memory Transactions
can take up to 600 clock
cycles!

Optimal reads from GPU global memory



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

Desktop					Mac Laptop	
N	Multi-CPU	1 GPU	3 GPUs	Base Time	GPU	Base Time
10^2	1x	2.5x	2.4x	5 mins	2.5x	8 mins
10^3	1x	14x	15x	30 mins	8x	50 mins
10^4	2x	60x	78x	5 hrs	15x	8 hrs
10^5	5x	93x	146x	2 days	16x	3 days
10^6	5x	100x	160x	22 days	25x	32 days

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

Desktop					Mac Laptop	
N	Multi-CPU	1 GPU	3 GPUs	Base Time	GPU	Base Time
10^2	1x	2.5x	2.4x	5 mins	2.5x	8 mins
10^3	1x	14x	15x	30 mins	8x	50 mins
10^4	2x	60x	78x	5 hrs	15x	8 hrs
10^5	5x	93x	146x	2 days	16x	3 days
10^6	5x	100x	160x	22 days	25x	32 days

Bottom Line: 22 days is reduced to 3 hours!

CPU versus GPU



The very alpha *fcm* library

- GPU code wrapped for use in R, Matlab and Python
- *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

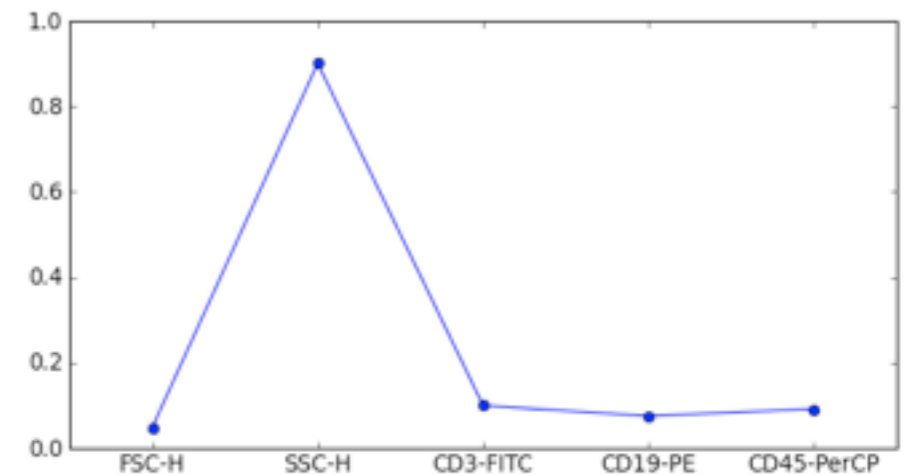
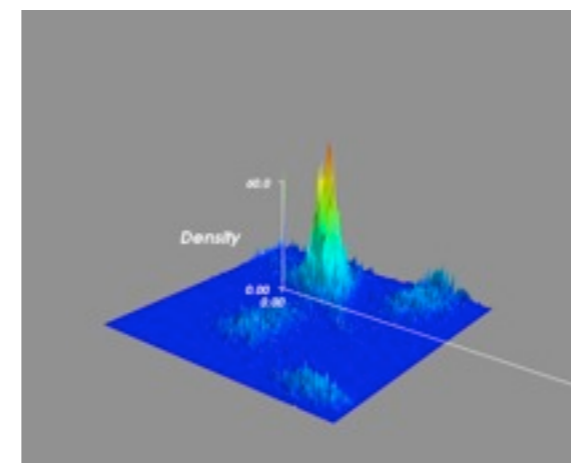
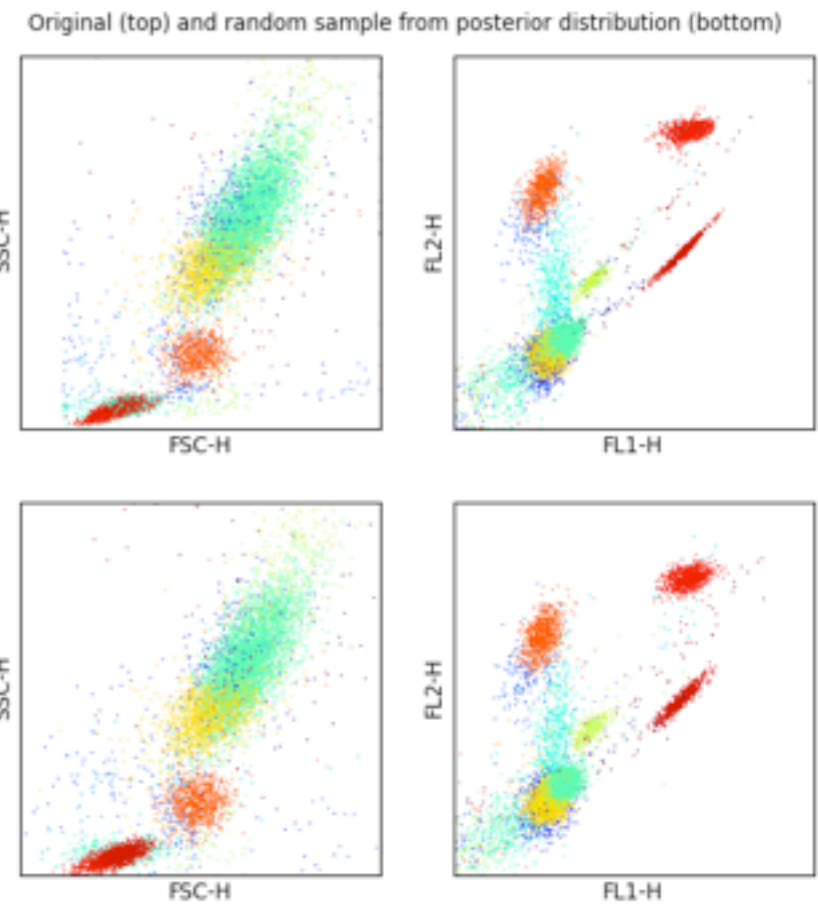
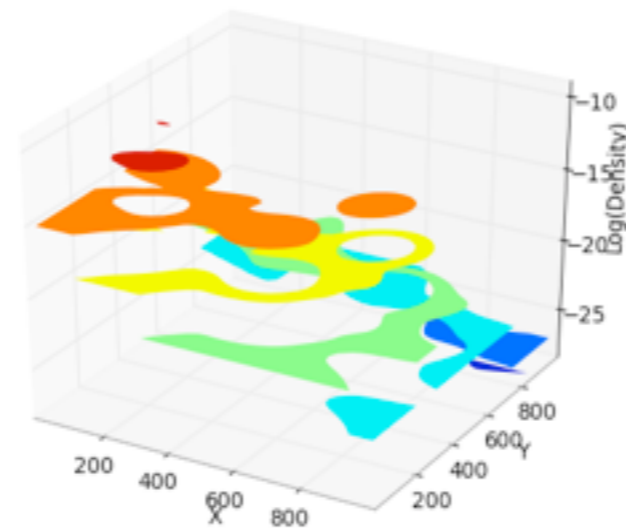
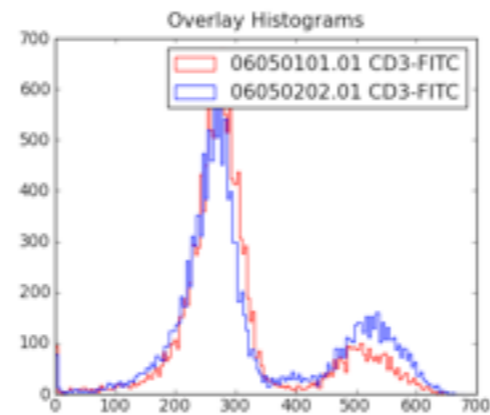
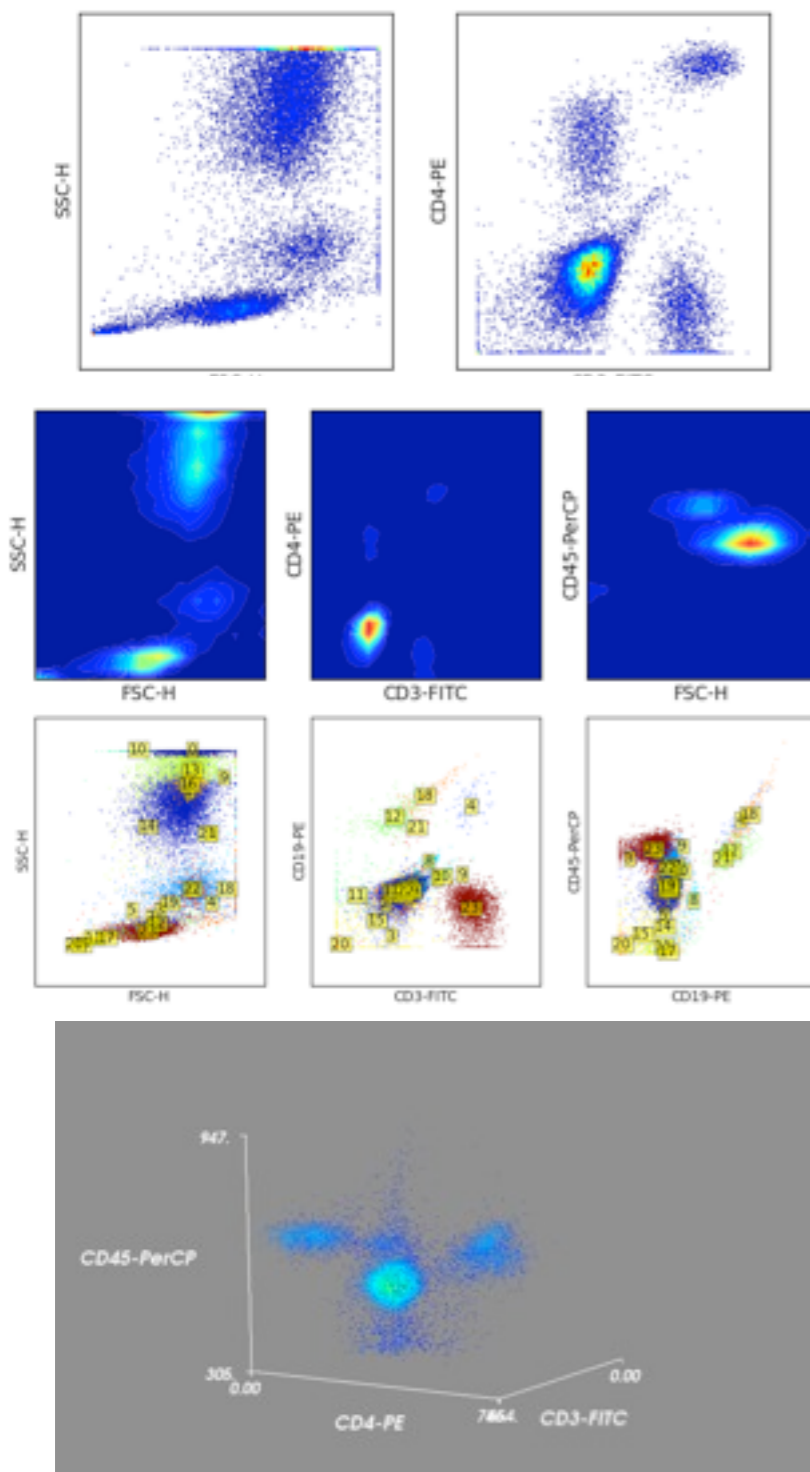
```
import fcm
import fcm.statistics as stats
import pylab

# 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')
```

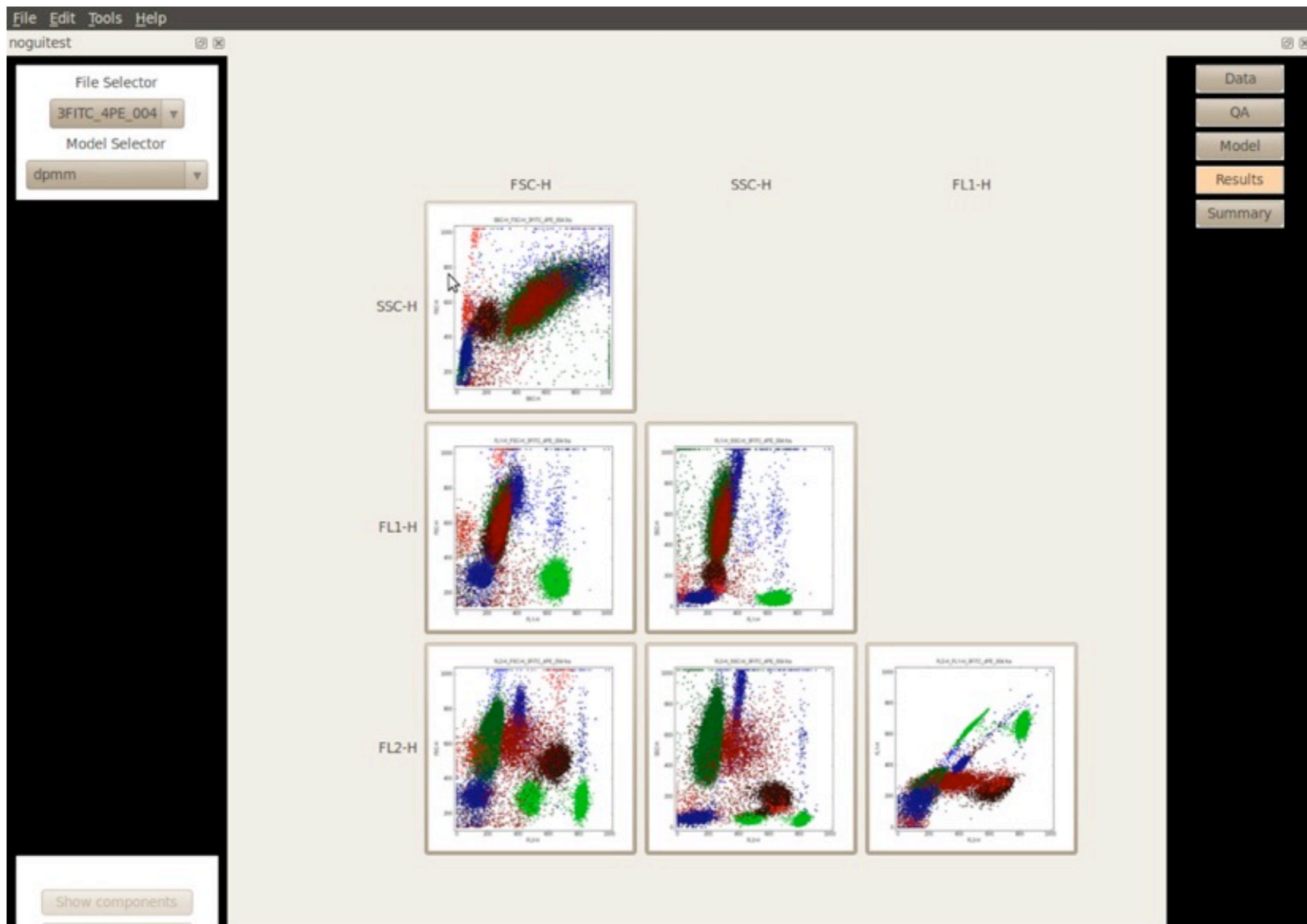
Some functionality

- 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

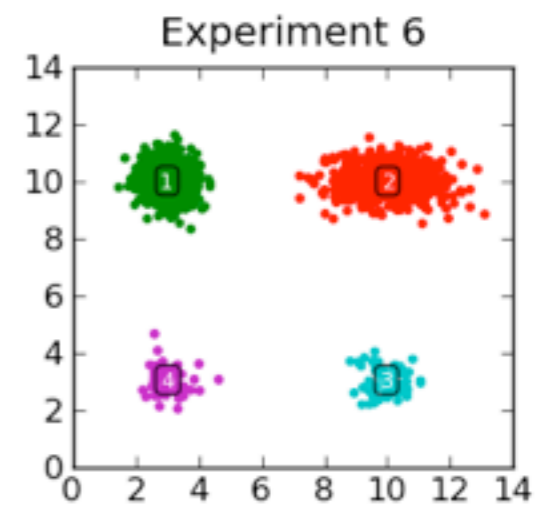
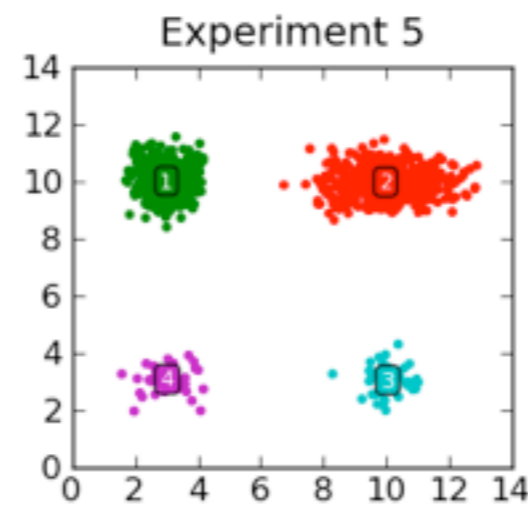
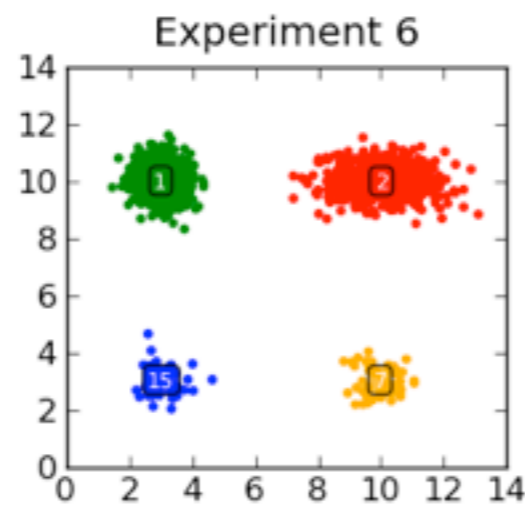
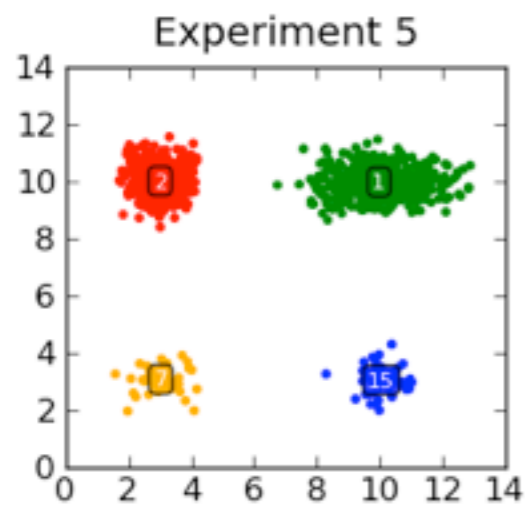
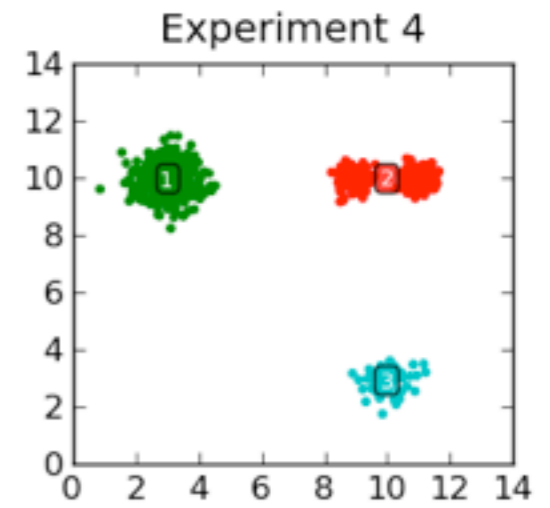
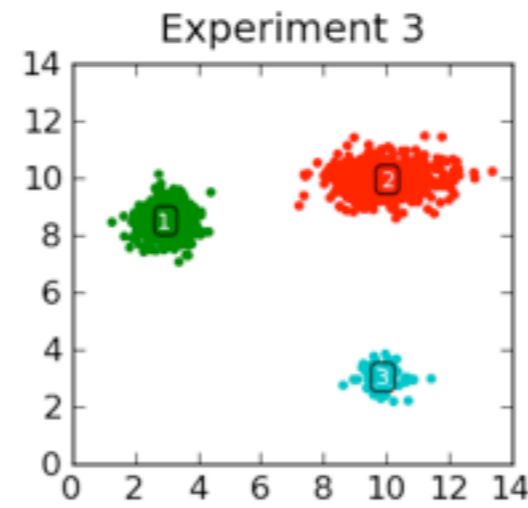
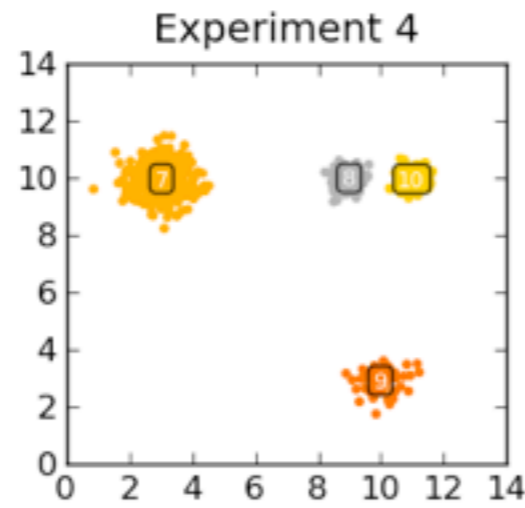
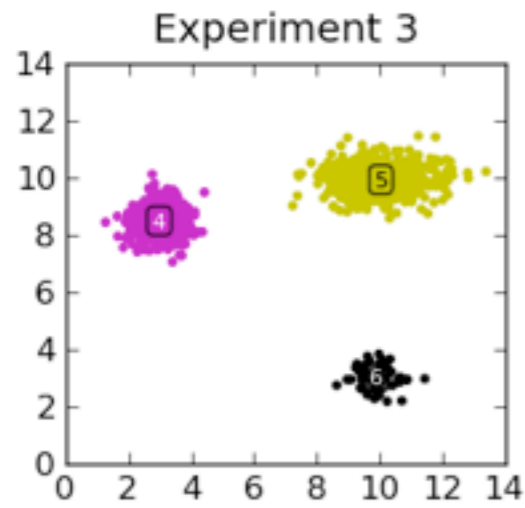
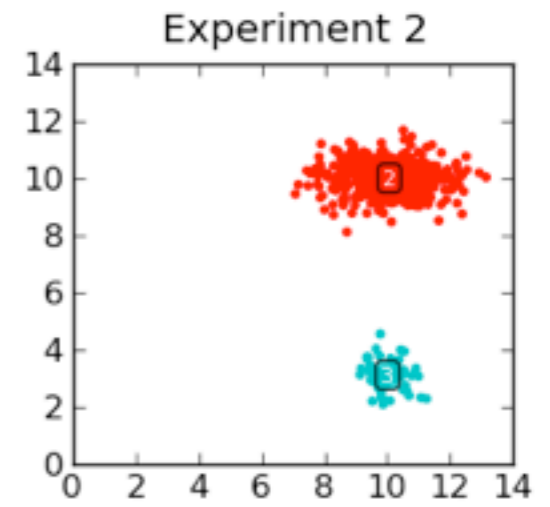
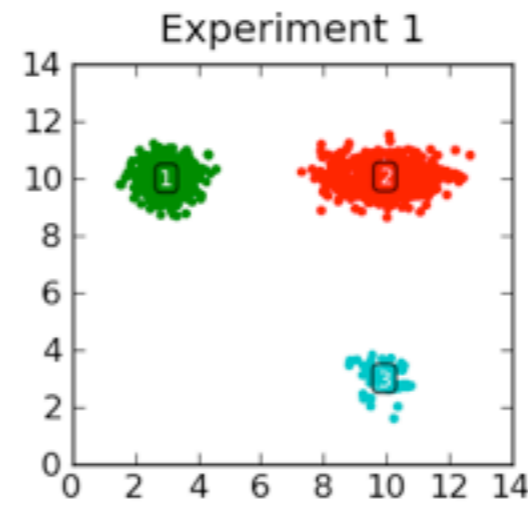
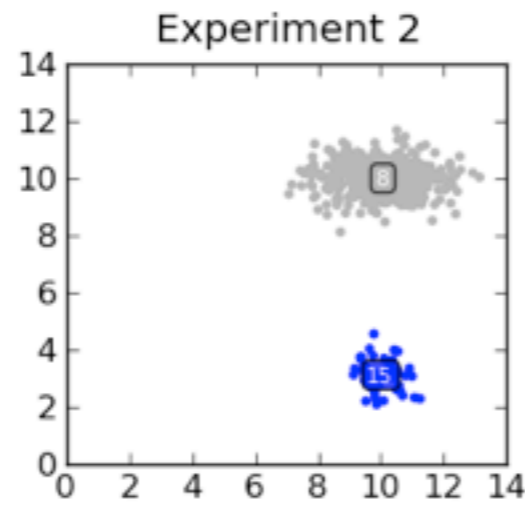
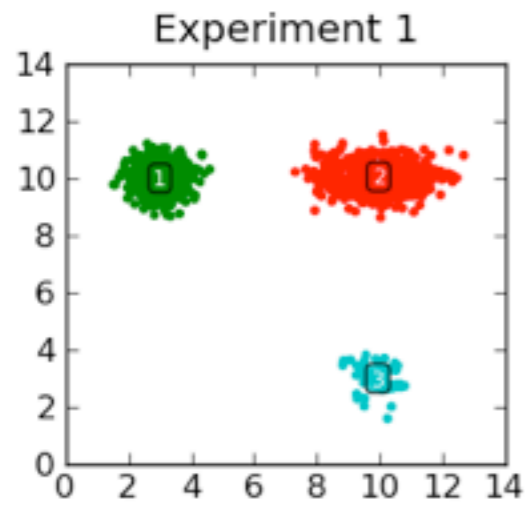


fcm in pipeline GUI



Before File Alignment

After File Alignment



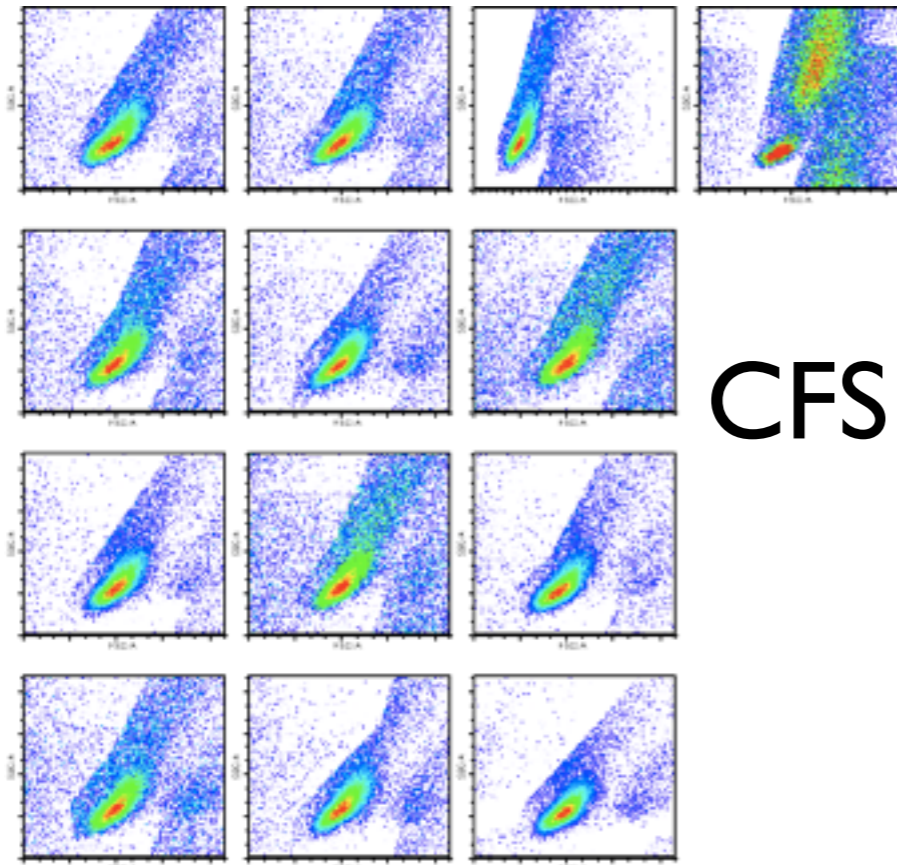
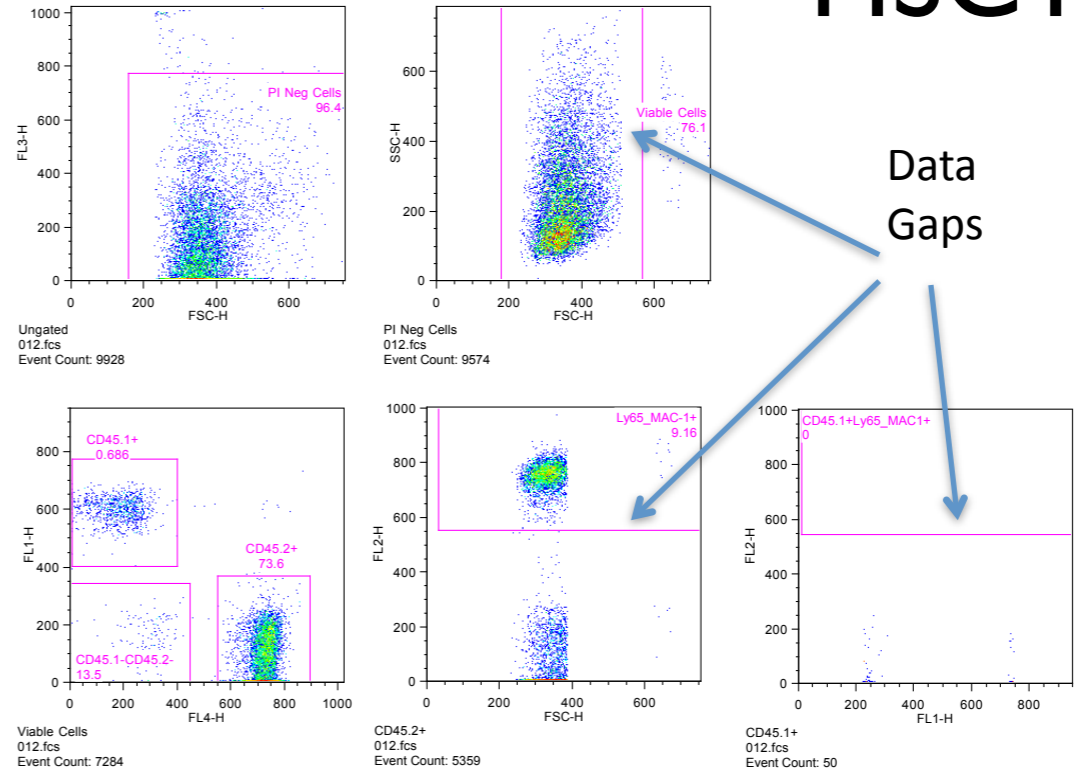
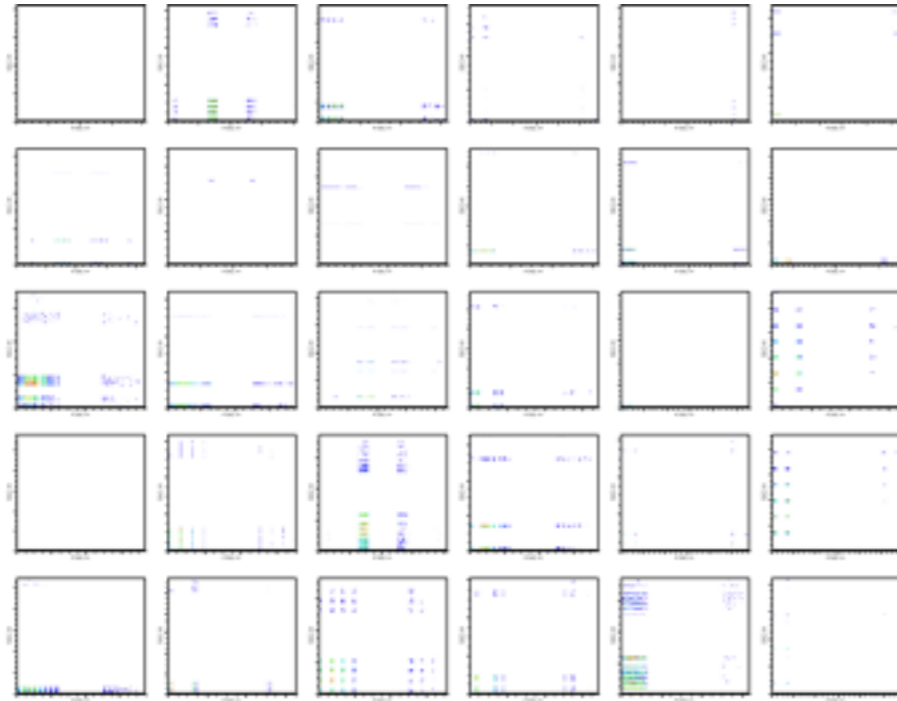
FlowCAP analysis

- 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

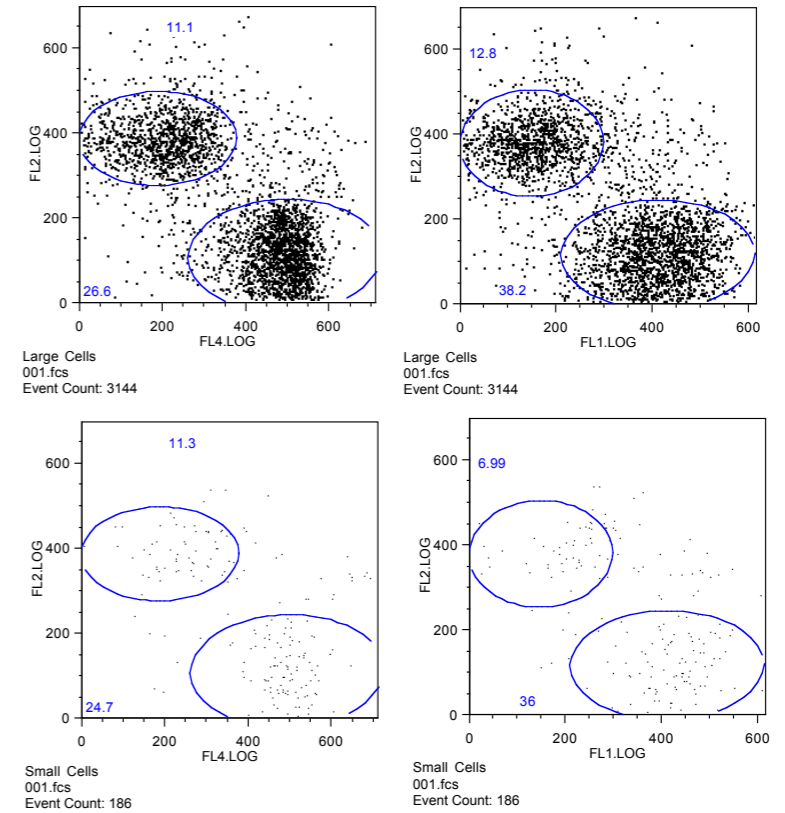
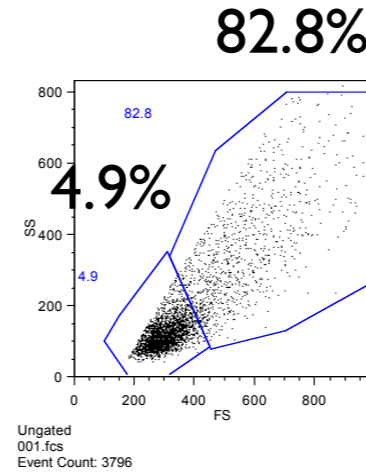
- Flow lab initially interested in helping us interpret data ... unfortunately, had lots of problems reading data with FlowJo and became frustrated instead

HSCCT scatters



CFSE

DLBCL



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, ULI-RR024128-04, RC1AI086032-01)