GPU-based Parallelized Quasi-random Parametric Expectation Maximization (QPREM) Estimation Algorithm for Population Data Analysis

Chee M Ng, PharmD, PhD, FCP

Outlines

- What is GPU and why do we want to use it?
- What is QRPEM and why QRPEM for GPU-computing?
- Example of first GPU-based QRPEM estimation method for population PK/PD data analysis

What is GPU

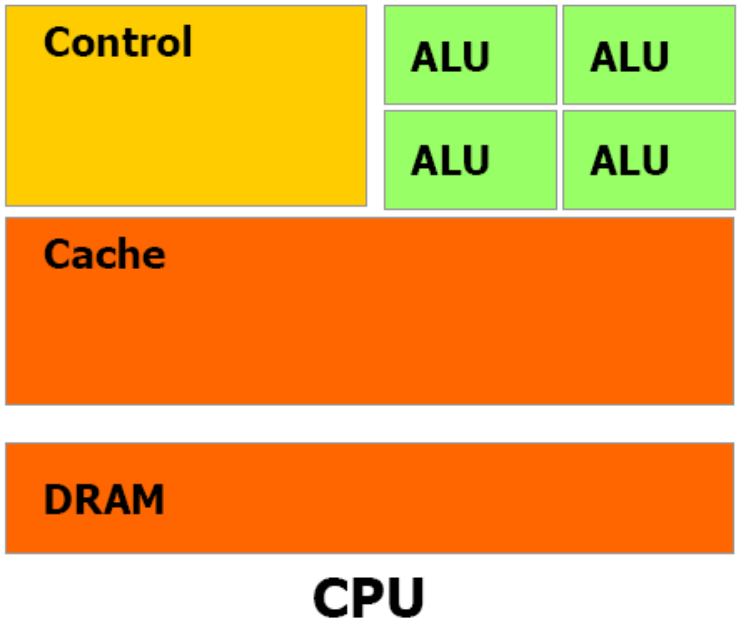
- **GPU= Graphic Processor Unit**
  - Chip in computer video cards, PlayStation 3, Xbox, etc.
  - Two major vendors: NVIDIA and ATI (AMD)
- Originally designed for maximum performance in numerical intensive image processing (modern games)
- GPUs are massively multithreaded many-core chips

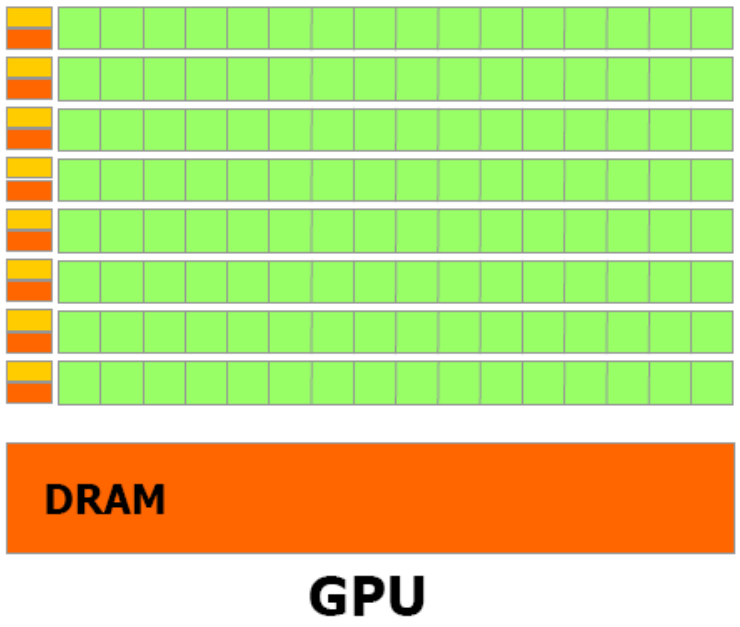
NVIDIA Quadro FX 5800 GPU card

* 240 parallel processing cores
* 930 GFLOPS sustained performances vs. 106 GFLOPS for Intel Core i7 975XE (3.3GHz)

Comparison Between Computer CPU and GPU

The GPU is specialized for compute-intensive, highly parallel computation
So, more transistor can be devoted to data processing rather than data caching and flow control





ALU – arithmetic logic unit that performs arithmetic and logical operations

Why GPU?

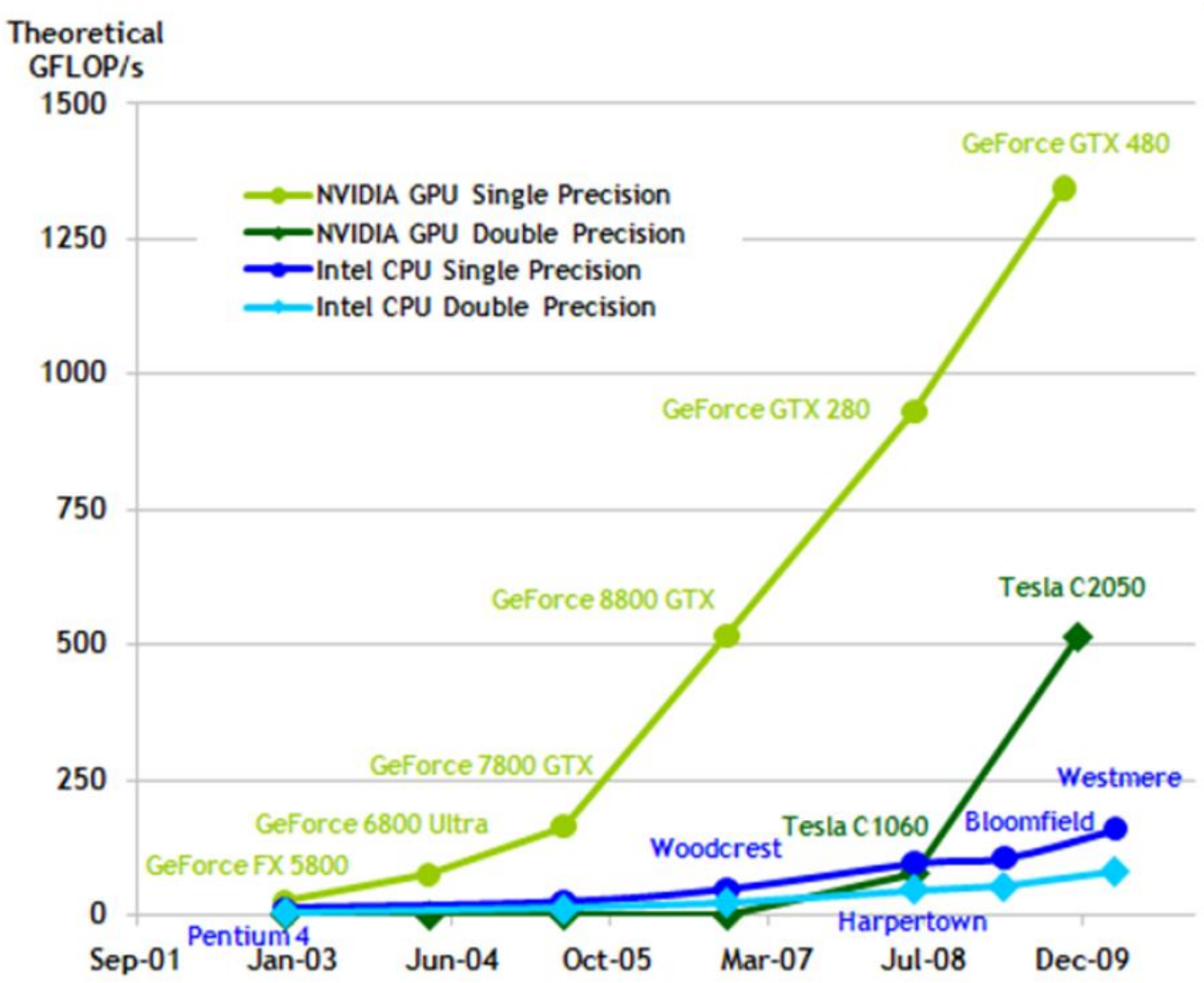
PRO

- Fast
- Cheap
- Low-power

CONS

- Specialized
- Hard to program
- Rapidly changing

GPU's are Much Faster Than CPU's



NVIDIA GeForce GTX 480
Cost ~ 300 USD

Source: CUDA Programming Guide; Intel Westmere : Intel Xeon X5600 series CPU

Fastest Supercomputer in the World is Powered by GPU Technology

Tianhe-1A system in China

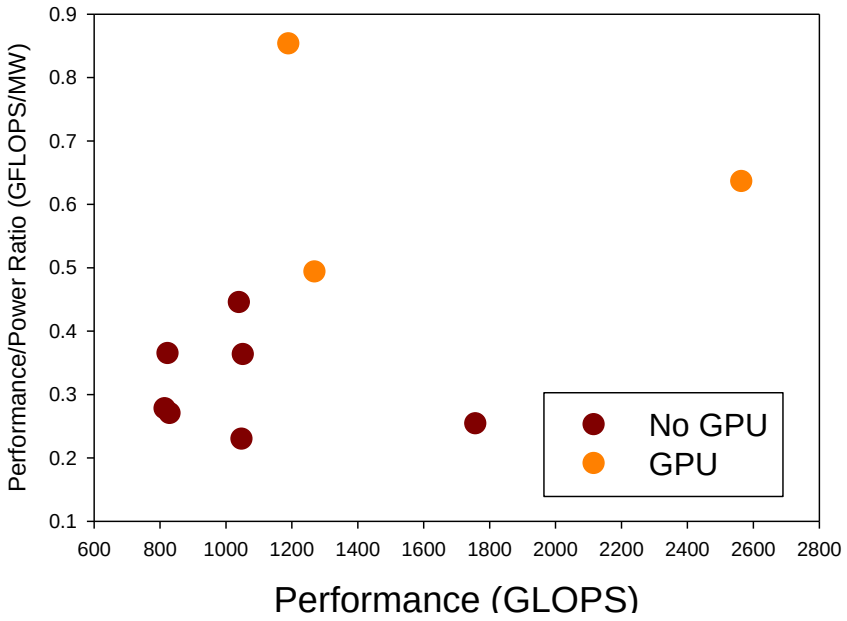
2.57 PFLOPS (10^{15} floating point calculations per second) !

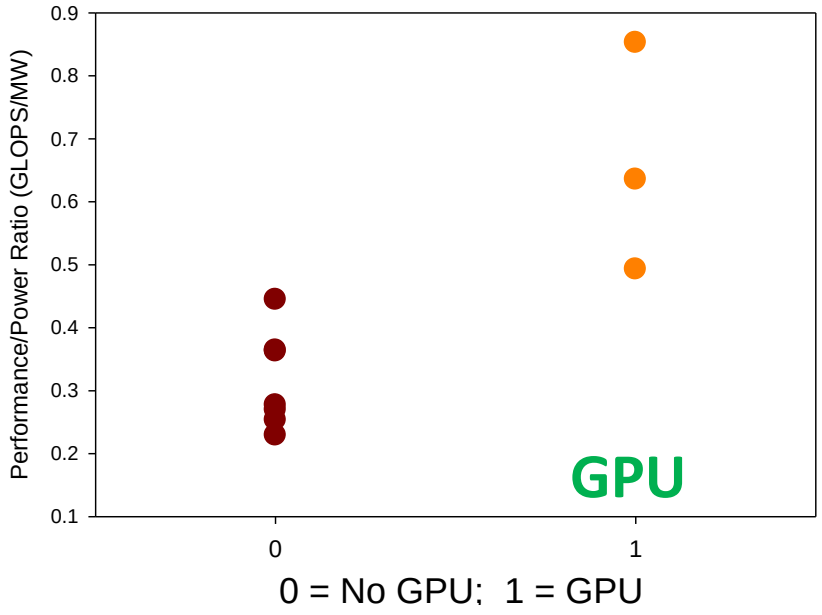
7,168 NVIDIA Tesla M2050 GPUs + 14, 366 CPUs

Source: http://www.top500.org/list/2010/11/100

Power Efficiency of the Supercomputer Performance/Power Ratio

Supercomputers powered by GPU-computing technology are more energy efficient **(GREEN- COMPUTING)**





Three of the World's Top Five Supercomputers are Powered by GPU-computing Technology

Data from Source: http://www.top500.org/list/2010/11/100

Why GPU?

PRO

- Fast
- Cheap
- Low-power

CONS

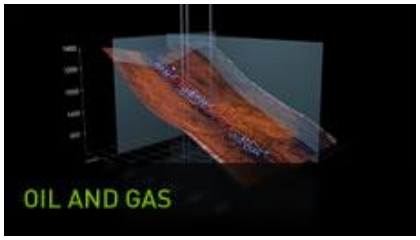
- Specialized
- Hard to program
- Rapidly changing

GPU Computing Today: CUDA

- **Compute Unified Device Architectures (CUDA)**
  - ***C programming language on GPUs***
  - Access to native instruction and memory
  - Requires no knowledge of graphic APIs or specific GPU programming
  - Developed by NVIDIA; Stable, available (for free), documented and supported for both Linux and Windows
  - Geared towards scientific programming
- GPU is now a "Computational Coprocessor"

Modified from http://www.scs.fsu.edu/~bollig/CUDA and John Seland: CUDA programming

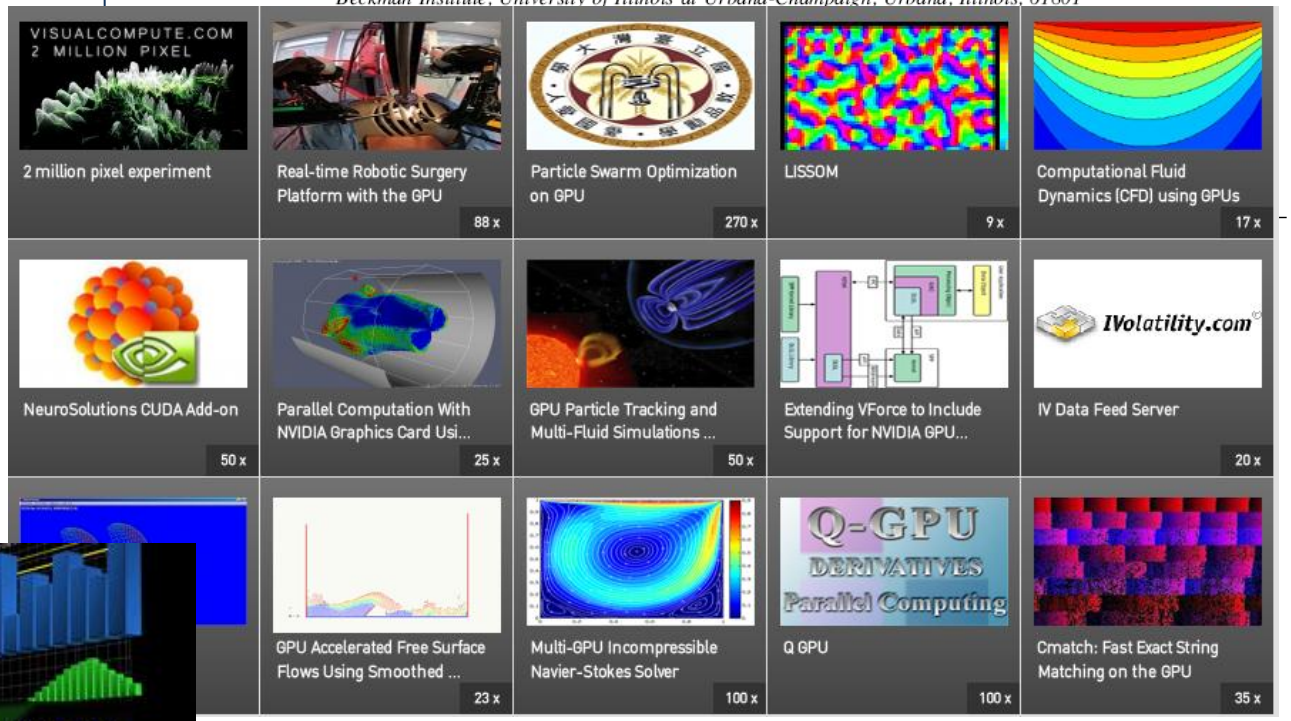
Successful Stories of GPU Computing



WILEY InterScience® DISCOVER SOMETHING GREAT

Accelerating Molecular Modeling Applications with Graphics Processors

JOHN E. STONE,[1*] JAMES C. PHILLIPS,[1*] PETER L. FREDDOLINO,[1,2*] DAVID J. HARDY,[1*]
LEONARDO G. TRABUCO,[1,2] KLAUS SCHULTEN[1,2,3]
[1]*Beckman Institute, University of Illinois at Urbana-Champaign, Urbana, Illinois, 61801*



Key words: GPU computing; CUDA; parallel computing; molecular modeling; electrostatic potential; multilevel

What is QRPEM and Why QRPEM for GPU Computing?

Two-stages (Parametric) NLME Estimation Methods Used in the Population PK/PD Data Analysis

- **Approximate Methods**
  - FO/FOCE (NONMEM) and ITS
- **Exact "Likelihood" Methods**

  Gaussian Quadrature and Importance Sampling

  EM – MCPEM (S-ADAPT and NONMEM), SAEM (Monolix, S-ADAPT and NONMEM), and QRPEM

NLME – Nonlinear mixed-effect model
FO – First-order; FOCE – First-order Conditional Estimation; ITS – Iterative 2-stages ;
SAEM - Stochastic Approximation EM; MCPEM – Monte-Carlo Parametric EM; QRPEM – Quassi-random Parametric EM – Parametric EM

Exact "Likelihood" Methods are Performed Better Than or Equal to the Methods That Approximated the Likelihood

Pascal Girard and France Mentré . A comparison of estimation methods in nonlinear mixed effects models using a blind analysis. **PAGE 14 (2005) Abstr 834**

The AAPS Journal 2007; 9 (1) Article 7 (http://www.aapsj.org).

Themed Issue: Bioinformatics and Computational Advances in the Pharmaceutical Sciences
Guest Editor - Murali Ramanathan

A Survey of Population Analysis Methods and Software for Complex Pharmacokinetic and Pharmacodynamic Models with Examples

Submitted: November 9, 2006; Accepted: January 13, 2007; Published: March 2, 2007

Robert J. Bauer,[1] Serge Guzy,[1] and Chee Ng[2]

[1]Pharmacokinetics, Pharmacodynamics, and Bioinformatics, XOMA (US) LLC, Berkeley, CA
[2]Institute for Drug Development/Cancer Research and Therapy Center, San Antonio, TX

EM-based "Exact-likelihood" Estimation Methods Were Used Successfully in Developing Population PK/PD Model

MCPEM

Pharmaceutical Research, Vol. 22, No. 7, July 2005 (© 2005)
DOI: 10.1007/s11095-005-5642-4

Research Paper

Pharmacokinetic–Pharmacodynamic–Efficacy Analysis of Efalizumab in Patients with Moderate to Severe Psoriasis

Chee M. Ng,[1,3] Amita Joshi,[1] Russell L. Dedrick,[2] Marvin R. Garovoy,[2] and Robert J. Bauer[2]

SAEM

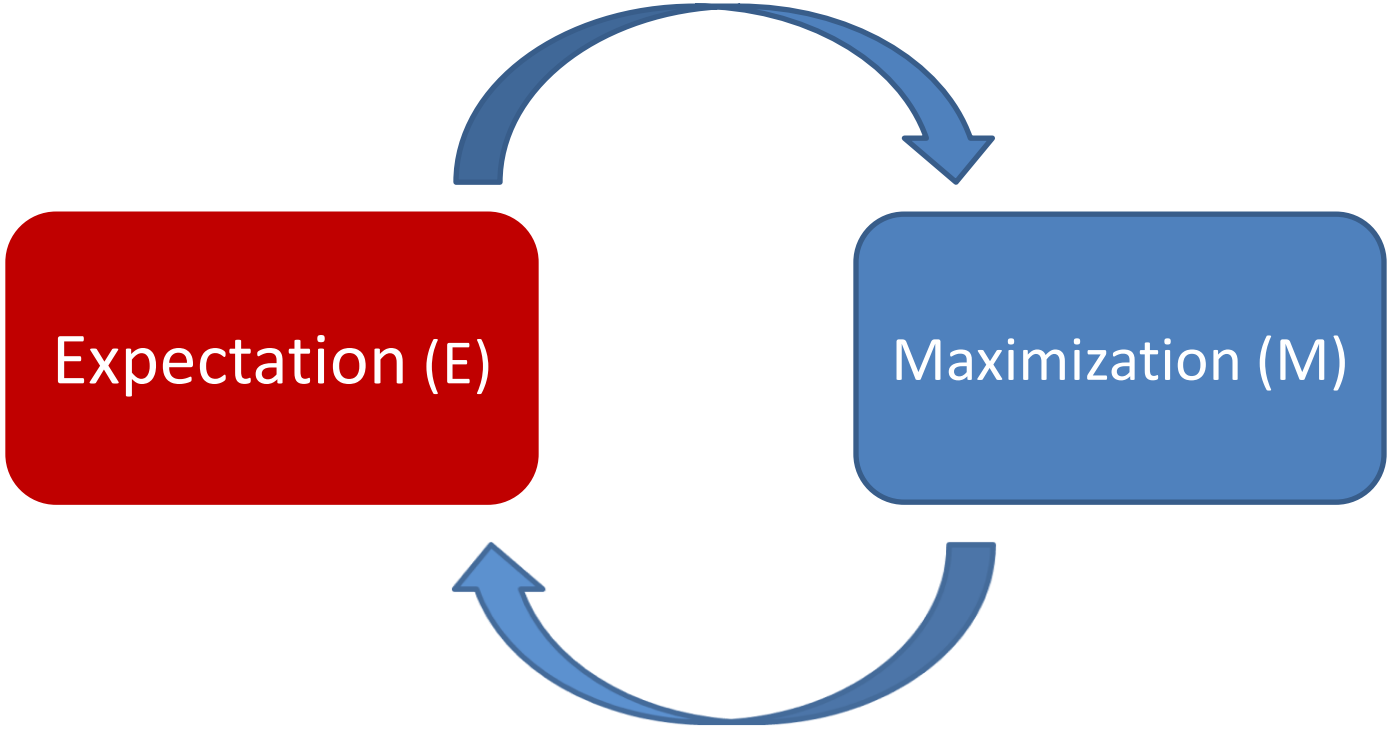
Estimation of Population Pharmacokinetic Parameters of Saquinavir in HIV Patients with the MONOLIX Software

Marc Lavielle[1,3] and France Mentré[2]

JPP 2007

Expectation Maximization (EM) Estimation Method for Population Data Analysis

- Iterative optimization process



Repeat E and M steps until population parameters no longer change (Maximum Likelihood is reached)

Expectation Maximization (EM) Algorithm: Expectation (E) Step

- The most computational intensive step in the EM
- Goal: to obtain individual conditional mean (mode) and variance-covariance matrix that used to update the population parameters in maximization (M) steps

Individual Conditional Mean

$$\overline{\theta_i} = \frac{\int_{-\infty}^{+\infty} \theta p(y_i, \theta \mid \mu, \Omega) d\theta}{\int_{-\infty}^{+\infty} p(y_i, \theta \mid \mu, \Omega) d\theta}$$

Expectation Maximization (EM) Algorithm: Maximization (M) Step

- Updating the population parameters

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\overline{\theta_i}$$

$$\Omega = \frac{1}{n}\sum_{i=1}^{n}(\overline{\theta_i} - \mu)(\overline{\theta i} - \mu)' + \frac{1}{n}\sum_{i=1}^{n}\overline{B_i}$$

μ= Population Mean; Ω = Population variance; θ_I = Individual conditional mean; B_i = individual variance-covariance matrix

EM Algorithm and Parallel Computing

- The EM algorithm is suitable for parallel computing because in the most computational intensive E step:
  - The conditional mean and variance of each subject
  - Generated random samples used to obtain the conditional mean and variance for each individual (Stochastic EM)
- *Are independent from each others,* and therefore can be evaluated separately!

EM Algorithm and Parallel Computing

The computation of the E step in the EM algorithm can be parallelized based on

1. **Subject** (Parallel computing of MCPEM in S-ADAPT/NONMEM)
2. **Generated random numbers within each subject** (GPU-based MCPEM)

First prototype of the GPU-based EM method (MCPEM using pseudo random number generator ; workstation with Tesla GPU) for population data analysis [ACOP 2011]

Penn
University of Pennsylvania

The Children's Hospital of Philadelphia
Hope lives here.

Novel GPU Parallelization of Monte-Carlo Parametric Expectation Maximization Estimation Algorithm for Population Data Analysis

Chee M Ng
Division of Clinical Pharmacology and Therapeutics, The Children's Hospital of Philadelphia; Philadelphia, PA
Department of Pediatric, School of Medicine, University of Pennsylvania, Philadelphia, PA

Classification of EM Estimation Methods for Population Data Analysis (Based on E-step)

- **Deterministic**
  - Gaussian Quadrature
- **Stochastic**
  - * **Sampling techniques**
    1. **Monte-Carlo**

       Direct Sampling (S-ADAPT), Rejection Sampling (SADAPT), **Importance Sampling (MCPEM in S-ADAPT/NONMEM),** Stratified Sampling, Recursive stratified sampling, VEGAS, and others
    2. **SAEM (MCMC) [Monolix/S-ADAPT/NONMEM]**
  - * **Random Number Generation**
    1. **Pseudo-random (PR)**
    2. **Quasi-random (QR)**

QR - PEM

QR – Quasi-random

The QR sampler can be used by many sampling techniques such as importance and direct sampling

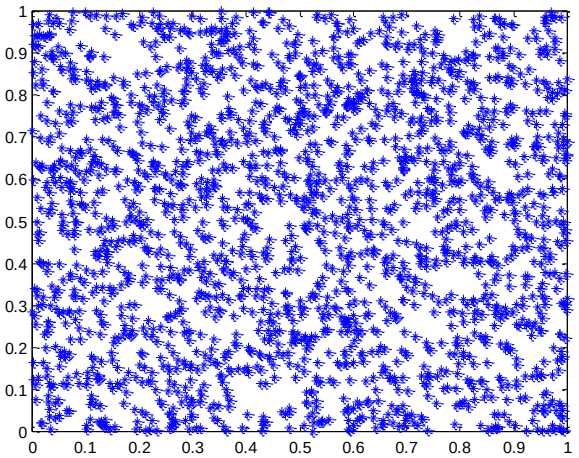
PEM – Parametric Expectation Maximization

Why QRPEM

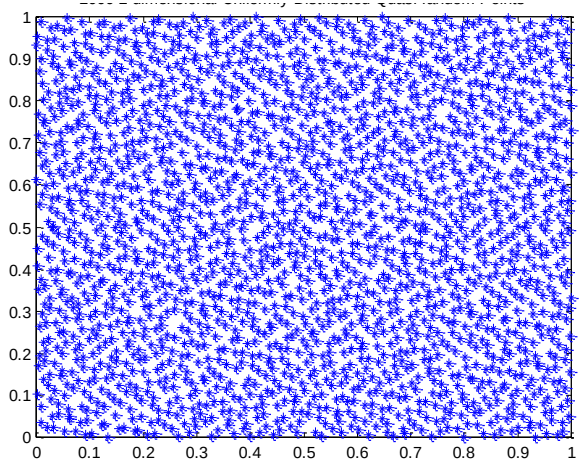
- Evaluation of the E-step in stochastic EM methods (MCPEM) required the computation of multi-dimensional integrals
- For pseudo-random (PR) number, the estimation error of the integrals will decrease at the rate of $N^{-1/2}$ (Error decay rate).
- Quasi-random (QR) sequence (low discrepancy sequences): In optimal case, QR has a much better decay rate of N^{-1}.
- To reduce the error by a factor of 10 → increase PR number by 100 x the number of simulation N, and in theory only needed ~ 10 X for QR

2000 2-dimensional random points

PR



QR



Source: Niederreiter 1992; Morokoff 2000; Birge 1995; Leary 2011

GPU-based QRPEM for Population PK/PD Data Analysis

- A single laptop computer equipped with an INTEL Core i7-920 Extreme Quad-core processor (2GHz) and
- NVIDIA Quadro FX3800M video graphic card with 128 stream processors

Windows 7 64-bit OS and 8GB RAM memory
NVIDIA FX3800M – 1G RAM; 60 GB/sec bandwidth (256-bit); clock speed - 675 MHz

GPU-based QRPEM Heterogeneous Computing

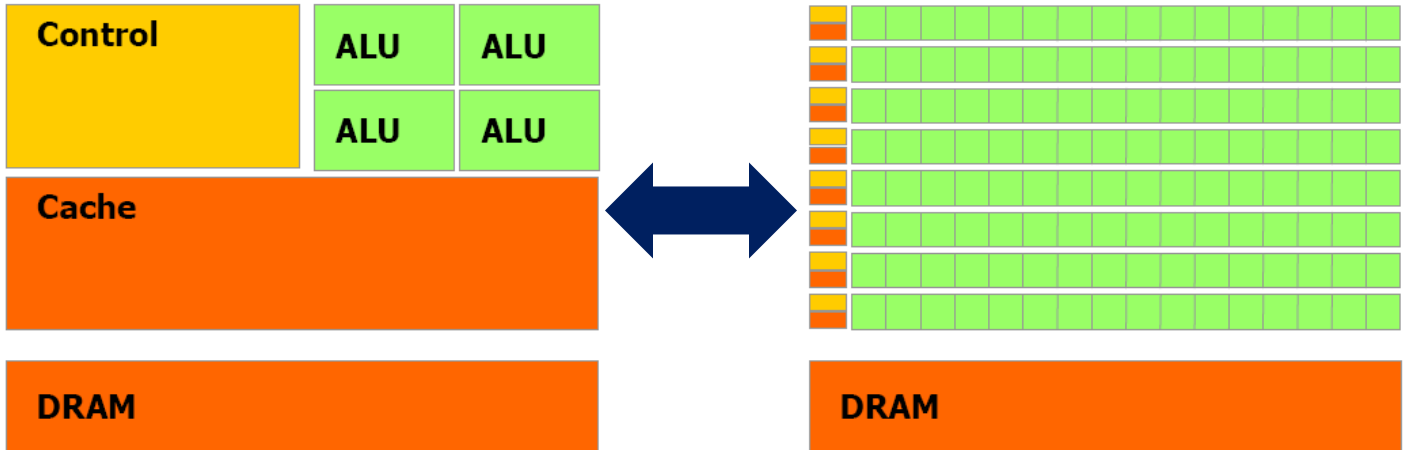
- **Computing with CPU and GPU**

CPU

M Step

GPU

E steps + partial derivatives of the intra-individual variance matrix



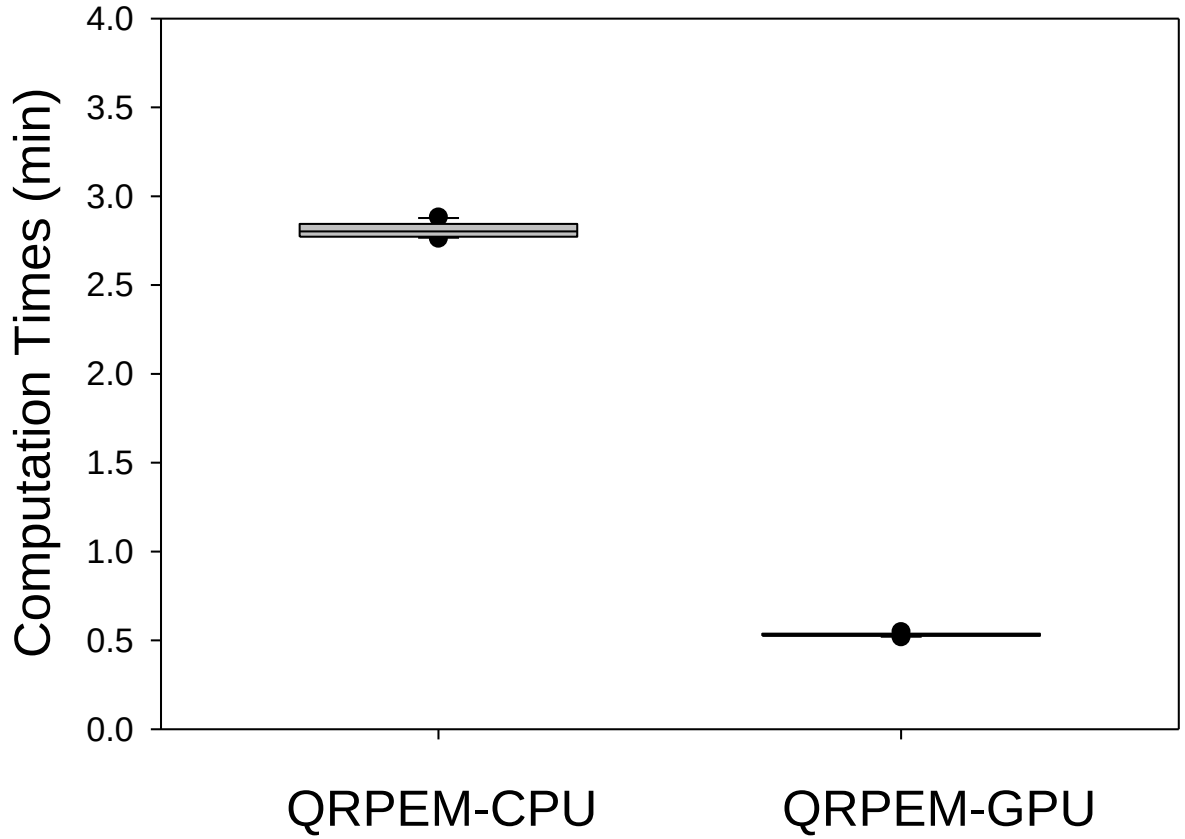
The GPU-based MCPEM (QRPEM-GPU) was developed using MATLAB R2009a and JACKET® GPU toolbox with NVIDIA CUDA GPU computing toolbox (3.2)

Simulated Data for Assessment of QRPEM-GPU Performance

- A one-compartment IV bolus PK model with intensive sampling schedule
  - Inter-subject variability: Log-normal distributed
  - Intra-subject variability: Proportional error model
  - Five system parameters (CL, V, IIV_CL, IIV_V and Sigma)

  Number of simulated trial = 100
- Number of simulated subjects for each trial: 25, 50, 100, and 150
- Number of QR (Sobol sequences with scrambling) direct random samples: 500, 1000, 2000, 5000, and 10000
- The results were compared to those obtained from a identical QRPEM method developed and executed in a INTEL CPU (QRPEM-CPU)

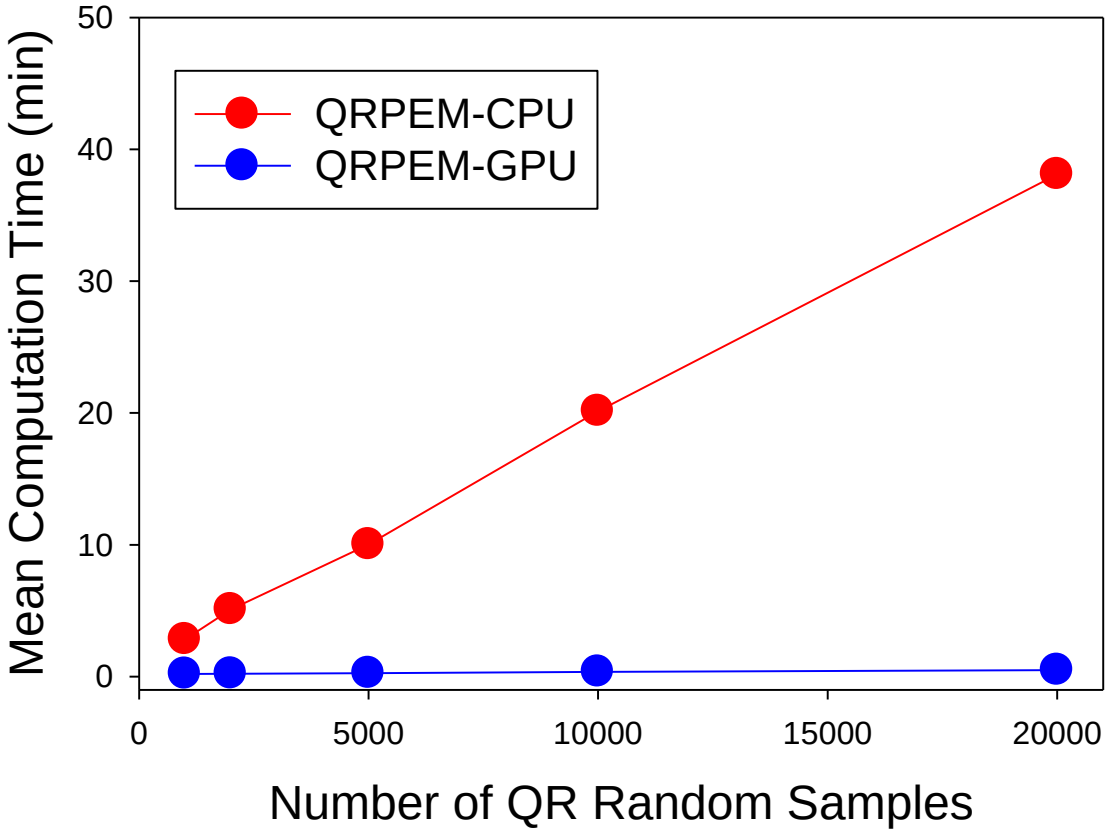
QRPEM-GPU Achieved Model Convergence Faster Than QRPEM-CPU



Number of Simulated Trial = 100; Number of simulated subject per trial = 100; Number of QR Samples for E step: 1000; Number of MCPEM iteration = 30

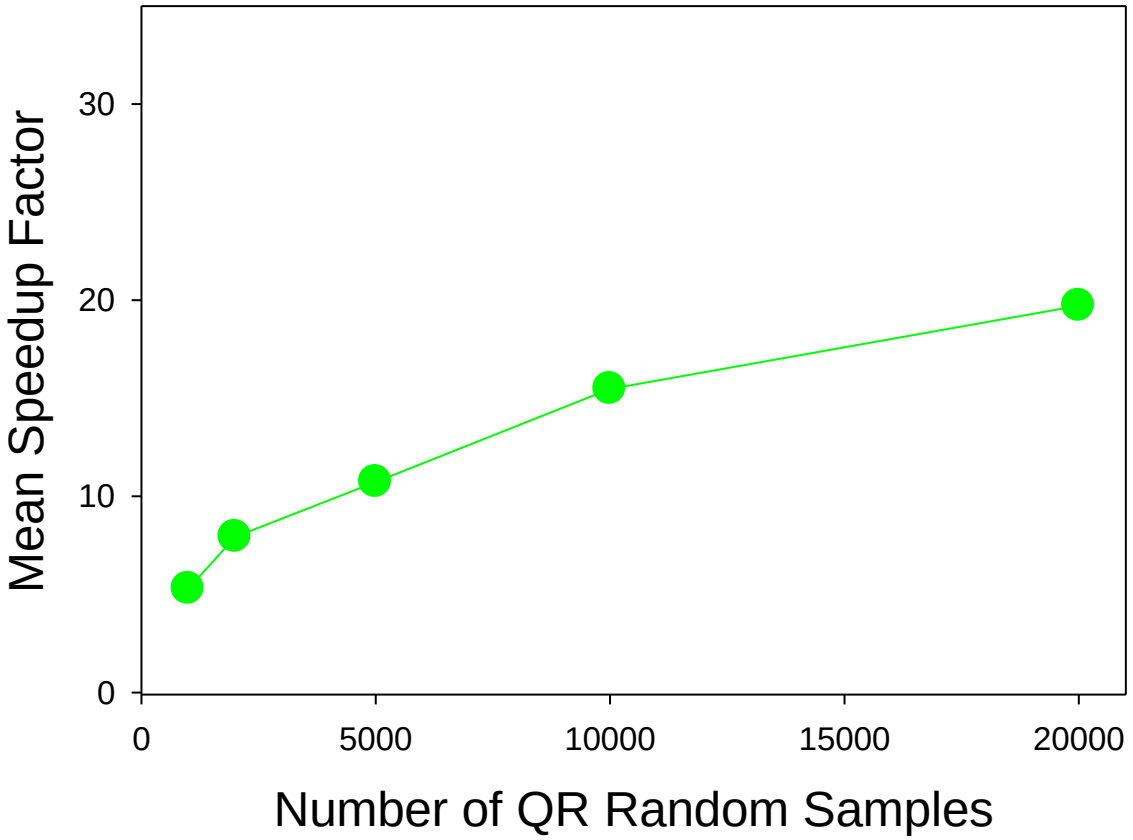
Performance of the QRPEM-GPU
Mean Model Converging Times vs. Number of QR Random Samples

- QRPEM-GPU has a better scaling relationships between mean model converging times and number of QR random samples



Speedup Factors of the QRPEM-GPU Increased in Proportional to the Number of Monte-Carlo Random Samples

QRPEM-GPU was ~20-folds faster than QRPEM-CPU in achieving model convergence when 20000 of QR random samples was used



Speedup factor = Model converging time of QRPEM-CPU/Model Converging Time of QRPEM-GPU

The Precision and Bias of the Final Model Parameters Were Comparable for Both QRPEM Algorithm

	CL	V	IIV_CL	IIV_V	Sigma
Precision (MAPE)					
QRPEM-CPU	4.9	5.3	3.0	5.1	2.3
QRPEM-GPU	4.9	5.3	3.0	4.7	2.2
Bias (MPE)					
QRPEM-CPU	4.9	5.3	0.36	-0.15	1.5
QRPEM-GPU	4.9	5.3	0.13	-0.29	1.5

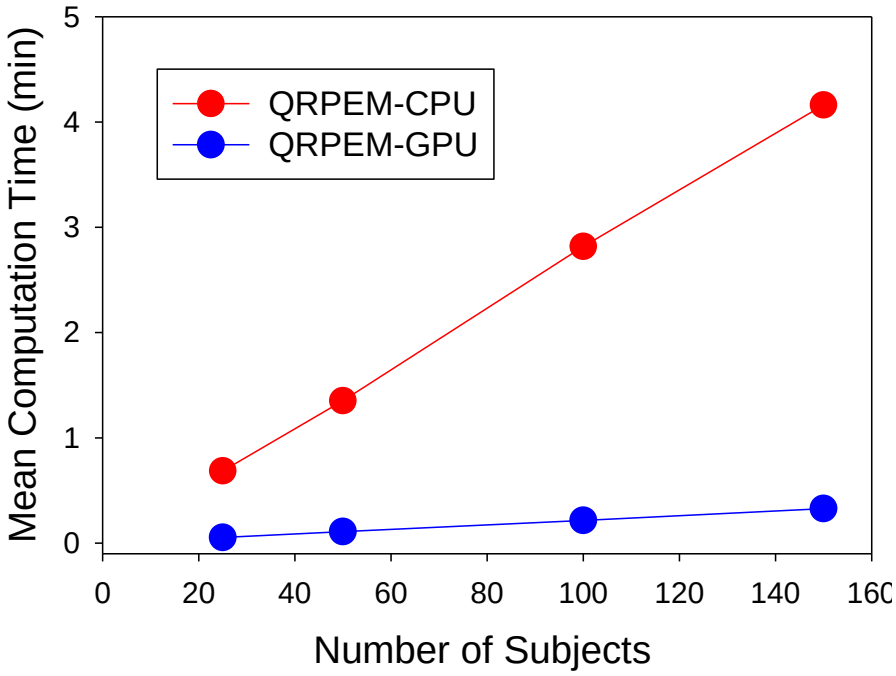
$$\mathrm{MAPE}=\frac{1}{n}\sum_{i=1}^{n}\left[\left|\frac{(\theta_i-\theta_{itrue})}{\theta_{itrue}}\right|\times 100\right] \qquad \mathrm{MPE}=\frac{1}{n}\sum_{i=1}^{n}\left[\frac{(\theta_i-\theta_{itrue})}{\theta_{itrue}}\times 100\right]$$

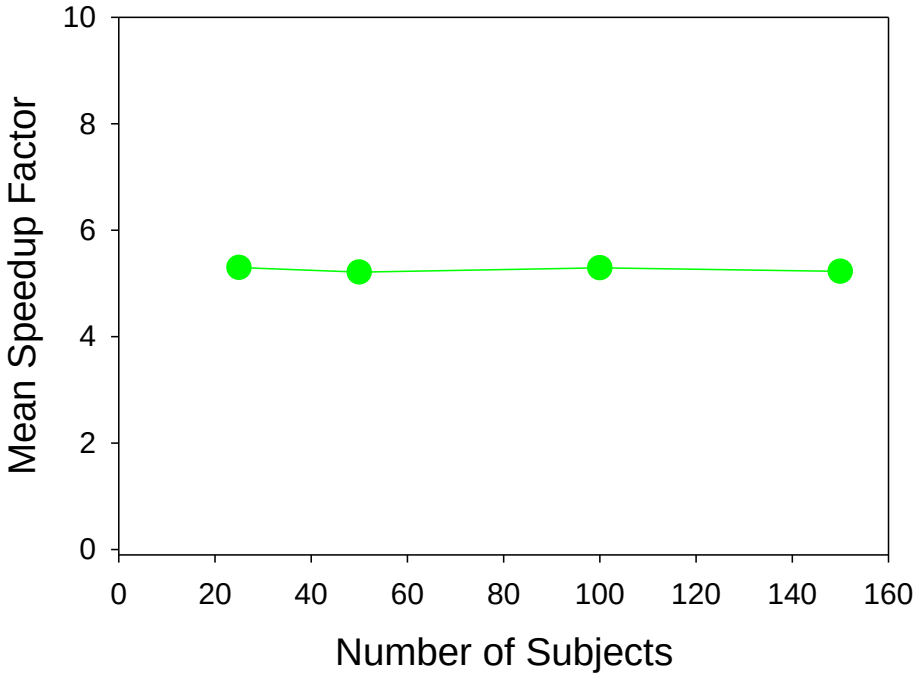
n: Number of simulated trials (=100); θ_i : Model estimated values; θi_{true}: True reference values

Number of Simulated Trial = 100; Number of simulated subject per trial = 100; Number of QR Samples for E step: 1000; Number of MCPEM iteration = 30

Performance of the QRPEM-GPU

Mean Model Converging Times vs. Number of Subjects





Number of Simulated Trial = 100; Number of QR Samples for E step: 1000; Number of MCPEM iteration = 30

Conclusions

- To my best knowledge, this is the first GPU-based parallelized QRPEM algorithm developed and reported in the literature for population PK data analysis
- Innovative, GPU-oriented approaches can lead to vast speed-up, and reduce data analysis and model development times

Future Works

- A study is ongoing to
  - expand the capability of the estimation algorithm in using parallel differential equation solver to develop complex population PK/PD model ; Multiple doses; Model converging criteria for likelihood ratio test
  - improve the efficiency of the algorithm either through further parallelization of the program codes or with multiple GPU processors

University of Pennsylvania/Children Hospital of Philadelphia NVIDIA CUDA Research Center

- Medical imaging analysis (DCI-MRI) in assessing the pharmacodynamic of the studied drug in preclinical/clinical studies
- GPU-based global optimization algorithm (GA/pattern-search) for complex PK/PD data analysis (Ng CM. ACOP 2010)
- GPU-based NLME Estimation method for population data analysis
- Machine learning/Artificial intelligent/Rule-based PK/PD/disease model development
- Others