cuDIMOT: A CUDA toolbox for modelling the brain tissue microstructure from diffusion-MRI

Moisés Hernández Fernández
Istvan Reguly, Mike Giles, Stephen Smith and Stamatios N. Sotiropoulos

GPU Technology Conference Europe 2017
Talk ID: 23165
Brain tissue microstructure

We want to gain information about tissue microstructure from diffusion MRI (dMRI) data:

- Understand the brain mechanisms
- Develop new biomarkers

Fibres dispersion

Fibres Orientation
1. Diffusion MRI

2. cuDIMOT: CUDA Diffusion Modelling Toolbox
   - Parallel design
   - Functionality and features

3. Results: Validation & Performance gains

4. Conclusions
diffusion MRI (dMRI)

- Molecules are in constant motion. We want to quantify water diffusion within a tissue.
- Different tissues:
  - Grey Matter: Diffusion without preferred direction.
  - White Matter: Diffusion along preferred direction.
- Information about tissue microstructure features can be gained getting several diffusion-weighted measurements and modelling the diffusion process using biophysical parameters.
Molecules are in constant motion. We want to quantify water diffusion within a tissue.

Different tissues:
- Grey Matter: Diffusion without preferred direction.
- White Matter: Diffusion along preferred direction.

Information about tissue microstructure features can be gained getting several diffusion-weighted measurements and modelling the diffusion process using biophysical parameters.
Molecules are in constant motion. We want to quantify water diffusion within a tissue.

Different tissues:
- Grey Matter: Diffusion without preferred direction.
- White Matter: Diffusion along preferred direction.

Information about tissue microstructure features can be gained getting several diffusion-weighted measurements and modelling the diffusion process using biophysical parameters.
cuDIMOT: Motivation

1200 Datasets

Subject Datasets

Tens to hundreds of measurements

Hundreds of thousands of voxels

- Nonlinear Optimisation
- Bayesian Inference

Complex diffusion MRI models with numerical approximations
cuDIMOT: Design

Model Designer

- Model Priors & Constraints
- Model Predicted Signal
- Model Partial Derivatives

cuDIMOT: CUDA Diffusion Modelling Toolbox

- Designer Interface
  - MCMC Bayesian Inference CUDA - Kernel
- User Interface
  - Grid-Search Least Squares CUDA - Kernel
  - Levenberg-Marquardt Least Squares CUDA - Kernel

Model User

Nifti File
cuDIMOT: Parallel design v1

Thread Block 0

\[
\begin{align*}
0 & 1 2 & \ldots & K-1 \\
\vdots & & & \\
0 & 1 2 & \ldots & K-1
\end{align*}
\]

Thread Block \((Vx*Vy/K) -1\)

\[
\begin{align*}
0 & 1 2 & \ldots & K-1 \\
\vdots & & & \\
0 & 1 2 & \ldots & K-1
\end{align*}
\]
cuDIMOT: Parallel design v2

Thread 0 (Leader)

m₀
m₃₂
m₆₄

Thread 1

m₁
m₃₃
m₆₅

Thread 2

m₂
m₃₄

... M measurements

Thread 31

m₃₁
m₆₃

Moisés Hernández Fernández, FMRIB
cuDIMOT: Parallel design v2

Thread Block 0

Vx VOXELS

Thread Block (Vx*Vy/B) -1

Vy VOXELS
cuDIMOT: Levenberg Implementation

**Levenberg**

- Cost function: sum of squared differences between measurements and model predictions
- Gradient descent method: needs partial derivatives for Gradient and Jacobian ($N_{Parameters} \times N_{Measurements}$)
- Threads collaborate for computing the partial derivatives
- $Hessian = Jacobian \times Jacobian^T$
- Shuffle instructions
- 2 warps (and voxels) per block

**Steps:**
1. Calculate partial derivatives for all the parameters.
2. Calculate Gradient & Jacobian & Hessian
3. LU solver
4. Compute model predicted signal & squared residuals
5. Calculate Cost function Accept/Reject step & Adapt $\lambda$
6. Synchronise
7. Synchronise
8. Synchronise

**Active Threads in a Block**

- 0 1 2 3 4
- 30 31

**Thread working**

**Idle thread**
A Generic toolbox

Options at compilation time:

- Bounds - Lower and/or Upper limits (any routine).
  Levenberg kernel implements reparameterisations internally

- Priors (MCMC):
  - Gaussian probability distribution
  - Gamma probability distribution
  - Automatic Relevance Determination (ARD)
  - Angle uniformly distributed on a sphere

- Constraints: relation between parameters

- Different noise models: Gaussian & Rician

- Numerical differentiation in Levenberg kernel
A Generic toolbox

Options at compilation time:

Model Predicted Signal $f(\theta) = \theta_1 \exp(-\theta_2 x)$ and Partial Derivatives

MACRO T Predicted_Signal (int npar, T* P, T* CFP, T* FixP){
    return P[0]*exp(-P[1]*CFP[0]);
}
MACRO void Partial_Derivatives (int npar, T* P, T* CFP, T* FixP, T* derivatives){
    derivatives[0]=exp(-P[1]*CFP[0]);
    derivatives[1]=-P[0]*CFP[0]*exp(-P[1]*CFP[0]);
}

Parameters Bounds and Priors

bounds[0] = (80,120)
bounds[1] = (.1,.5)
prior[0] = Gaussian(100,10)
prior[1] = ARD(1)
A Flexible toolbox

Functionality at execution time:

- Choosing fitting routines: Grid search, Levenberg-Marquardt, MCMC
- Selecting number of iterations in Levenberg-Marquardt
- Selecting number of iterations in MCMC (burn-in, total, sample thinning interval)
- Cascaded model fitting (Initialising parameters)
- Choose parameters of the model to be kept fixed during the fitting process
- Bayesian & Akaike Inference Criterion

The toolbox can be easily extended
Validation: Fibre Orientation

We have implemented several diffusion models

Fibre Orientation estimation:

- Superior - Inferior
- Anterior - Posterior
- Medial - Lateral
- Coronal
- Sagittal
- Axial

CPU

GPU cuDIMOT
Validation: Fibre Orientation

Mean estimates: 1000 repeats
Validation: Fibre Orientation

Standard deviation estimates: 1000 repeats

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Callosum</th>
<th>Centrum</th>
<th>Semiovale</th>
<th>Grey Matter</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0$</td>
<td>$d$</td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>Orientation Uncertainty</td>
</tr>
<tr>
<td>CPU</td>
<td>GPU cuDIMOT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 ... 1</td>
<td>0 ... 1</td>
<td>0 ... 1</td>
<td>0 ... 1</td>
<td>0 ... 1</td>
</tr>
<tr>
<td>0.00002</td>
<td>0.00004</td>
<td>0.00004</td>
<td>0.00002</td>
<td>0.00002</td>
</tr>
<tr>
<td>0.00006</td>
<td>0.00008</td>
<td>0.00006</td>
<td>0.00004</td>
<td>0.00004</td>
</tr>
<tr>
<td>0.00004</td>
<td>0.00006</td>
<td>0.00004</td>
<td>0.00002</td>
<td>0.00002</td>
</tr>
<tr>
<td>0.00002</td>
<td>0.00004</td>
<td>0.00004</td>
<td>0.00002</td>
<td>0.00002</td>
</tr>
<tr>
<td>0.00005</td>
<td>0.00010</td>
<td>0.00010</td>
<td>0.00005</td>
<td>0.00005</td>
</tr>
<tr>
<td>0.00003</td>
<td>0.00006</td>
<td>0.00006</td>
<td>0.00003</td>
<td>0.00003</td>
</tr>
<tr>
<td>0.00002</td>
<td>0.00004</td>
<td>0.00004</td>
<td>0.00002</td>
<td>0.00002</td>
</tr>
</tbody>
</table>
Validation: Fibre Dispersion

Moisés Hernández Fernández, FMRIB

3. Results: Validation
Validation: Fibre Dispersion

3. Results: Validation

- **White Matter**
  - NODDI-Watson vs. cuDIMOT
  - Pearson CC: 0.998

- **Grey Matter**
  - NODDI-Watson vs. cuDIMOT
  - Pearson CC: 0.997, 0.989, 0.977

- **White & Grey Matter**
  - NODDI-Watson vs. cuDIMOT
  - Pearson CC: 0.997, 0.995, 0.992
Performance

Time fitting different dMRI models

<table>
<thead>
<tr>
<th>Model</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODDI Watson Matlab</td>
<td>352x</td>
</tr>
<tr>
<td>NODDI Bingham Matlab</td>
<td>6.98x</td>
</tr>
<tr>
<td>Ball &amp; 1 Stick Matlab</td>
<td>3.85x</td>
</tr>
<tr>
<td>Ball &amp; 1 Stick C++</td>
<td>3.88x</td>
</tr>
<tr>
<td>Ball &amp; 2 Sticks Matlab</td>
<td>3.26x</td>
</tr>
<tr>
<td>Ball &amp; 2 Sticks C++</td>
<td>3.7x</td>
</tr>
</tbody>
</table>

Times in seconds (logarithm scale)

CPU tool - 72 cores

cuDIMOT - single K80 GPU

3. Results: Performance gains
Performance limitations

Global Memory Bandwidth

- Reads: 95.23 GB/s
- Writes: 61.50 GB/s
- Total: 156.73 GB/s
- ECC Overhead: 40.36 GB/s

Global Memory Load Efficiency: 91.4%

Global Memory Store Efficiency: 90.5%
Conclusions

- Diffusion MRI allows the study of brain microstructure non-invasively and in-vivo, but it can be very time-consuming.

- cuDIMOT: We have designed and implemented a generic and flexible CUDA toolbox for nonlinear model fitting (new models can easily be implemented on GPUs). It reduces computational times: $\sim 200X$.

- These accelerations are tremendously beneficial, especially in very large recent studies such as:
  - The Human Connectome Project (HCP): data from 1,200 adults
  - The UK Biobank Project: data from 100,000 adults.

- cuDIMOT can be used in other modalities and can be easily extended
Acknowledgements

Funding:

- Human Connectome Project (1U54MH091657-01)
- UK EPSRC (EP/L023067/1)