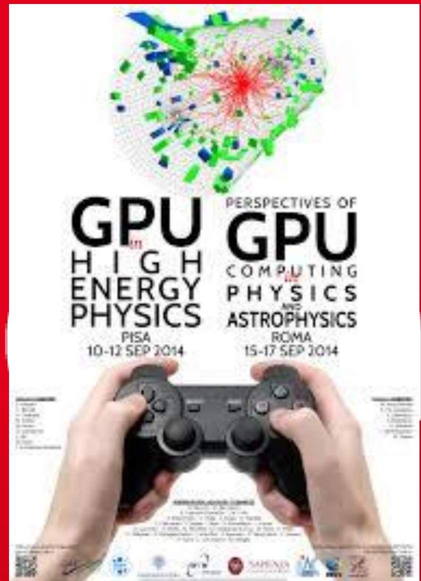


GPU-PARALLELIZED LEVENBERG-MARQUARDT MODEL FITTING TOWARDS REAL-TIME AUTOMATED PARAMETRIC DIFFUSION NMR IMAGING

Marco | Palombo[1,2],

D. Zhang[3], X. Zhu[4], J. Valette[1], A. Gozzi[5], A. Bifone[5], A. Messina[6], G. Lamanna[7], S. Capuani[2]

1| MIRCen, I^{2}BM, DSV, CEA Fontenay-aux-Roses, France

2| IPCF-UOS Roma, Physics Dpt., Sapienza University, Rome, Italy

3| ITG, Beckman Institute for Advanced Science & Technology, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA

4| College of Economics & Management, China Agricultural University, Beijing, China

5| IIT Center for Neuroscience and Cognitive Systems @ UniTn, Rovereto, Italy

6| Physics Dpt. Sapienza University, Rome, Italy

7| National Institute of Nuclear Physics, Pisa Section, Pisa, Italy

12 SEPTEMBER 2014

OUTLINE

In this contribution we report one of the main goals of the **G**PU **A**pplication **P**roject (GAP), which aims to investigate the deployment of Graphic Processing Units (GPU) in different context of realtime scientific applications.

In particular we focused on the application of GPUs in reconstruction of diffusion weighted nuclear magnetic resonance (DW-NMR) images by using non-Gaussian diffusion models.

OUTLINE

- State of the art of non-Gaussian DW-NMR imaging
- Why and how using GPU to make DW-NMR imaging faster
- Application: non-Gaussian DW-NMR imaging of mouse brain on GPU
- Future perspectives about GPU applications in NMR
- Conclusion

Typical DW-NMR imaging post-processing

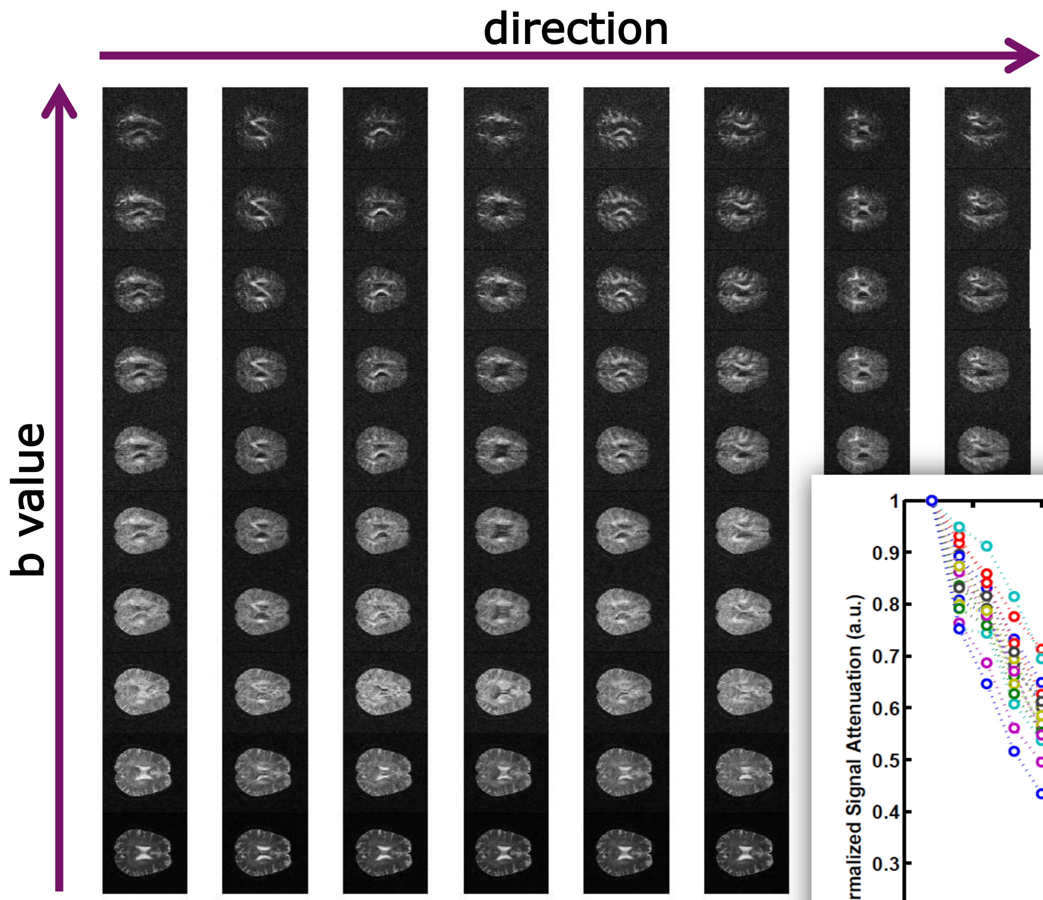
$$Signal(\boldsymbol{k},t) \propto \int P(\boldsymbol{r},t) e^{-i\boldsymbol{k}\cdot\boldsymbol{r}} d\boldsymbol{r}$$

Weight of the NMR signal in diffusion (**directional**)

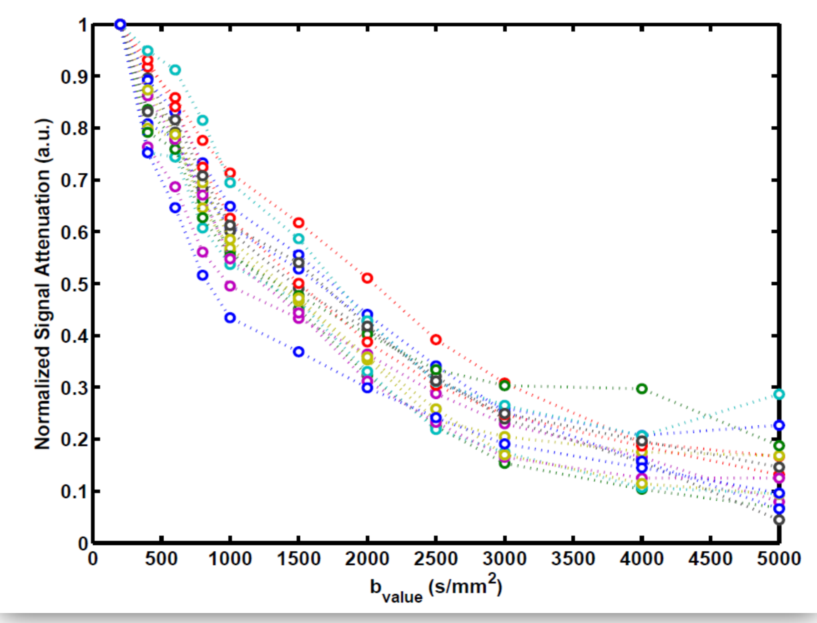
$$\boldsymbol{b} = k^2 t\, \widehat{\boldsymbol{k}}$$

- In **homogeneous** media P(r,t) is **Gaussian**
- In **heterogeneous** complex media P(r,t) is **not Gaussian**

STATE OF THE ART OF DW-NMR IMAGING

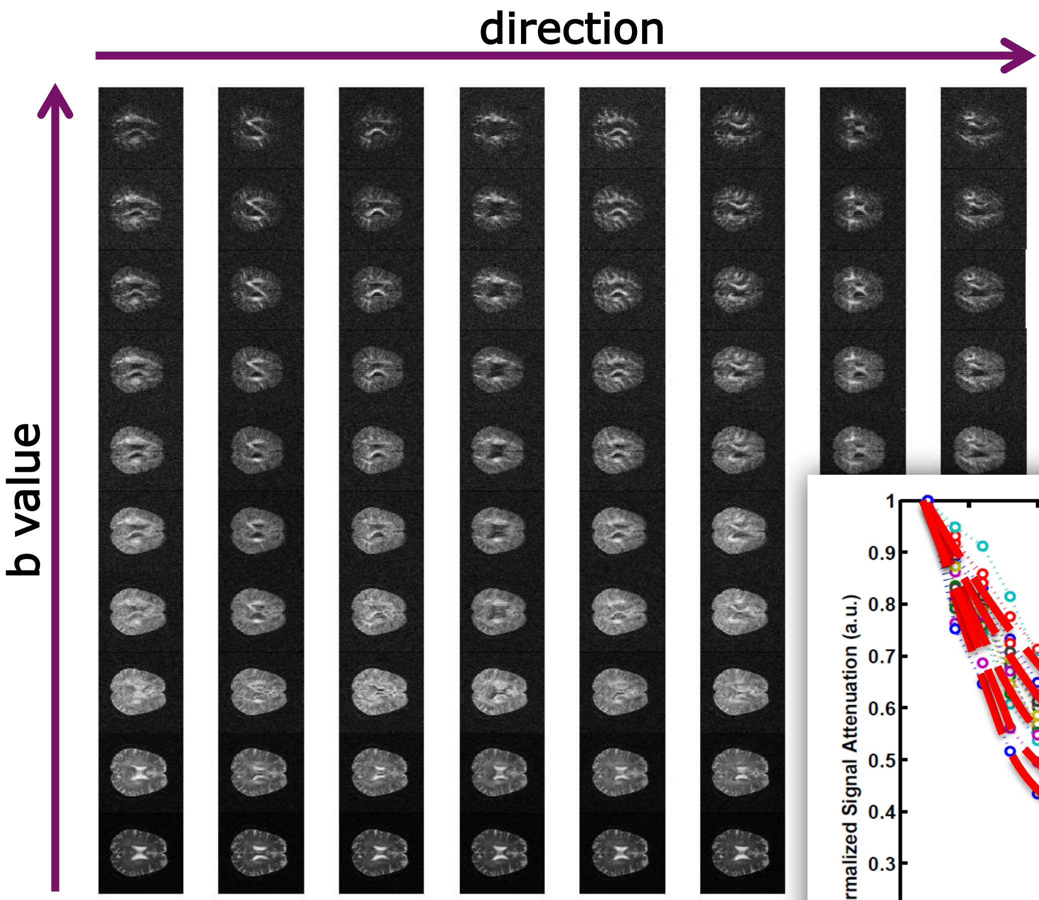


Typical DW-NMR imaging post-processing

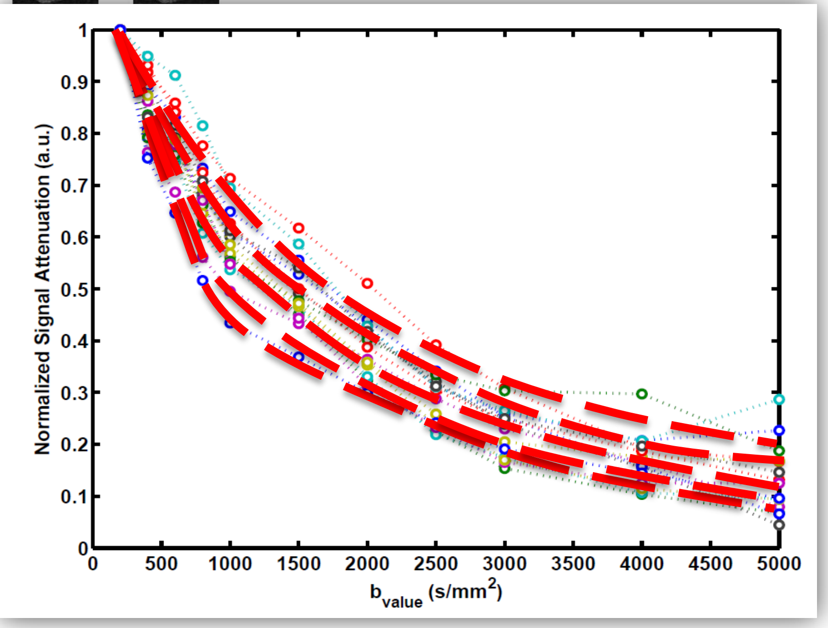


128x128 pixel image
x
number of slice (~10-40)
x
number of diffusion gradient directions (≥15)

STATE OF THE ART OF DW-NMR IMAGING

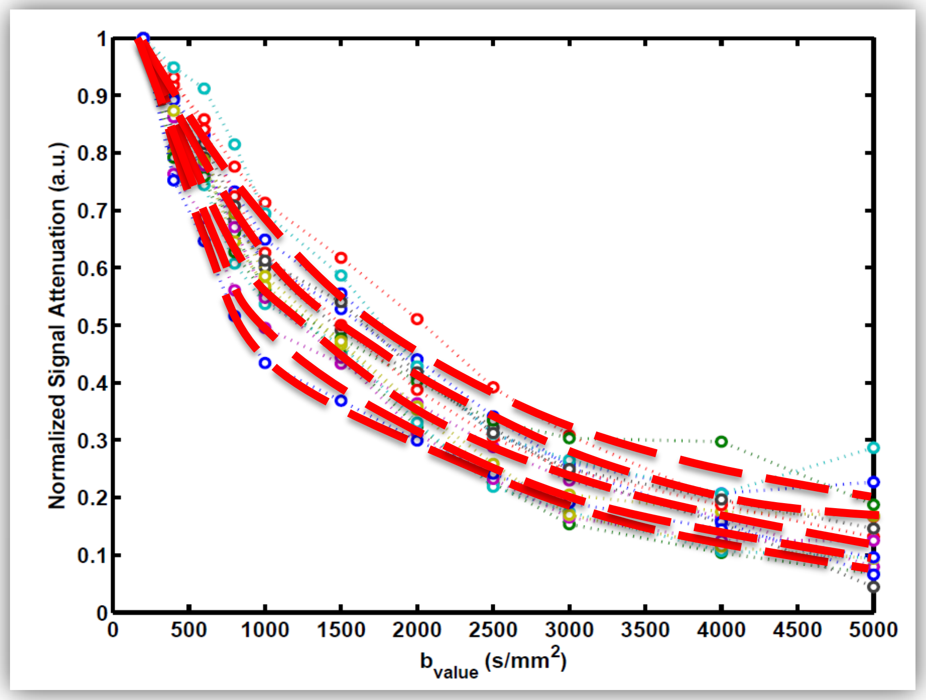


Typical DW-NMR imaging post-processing



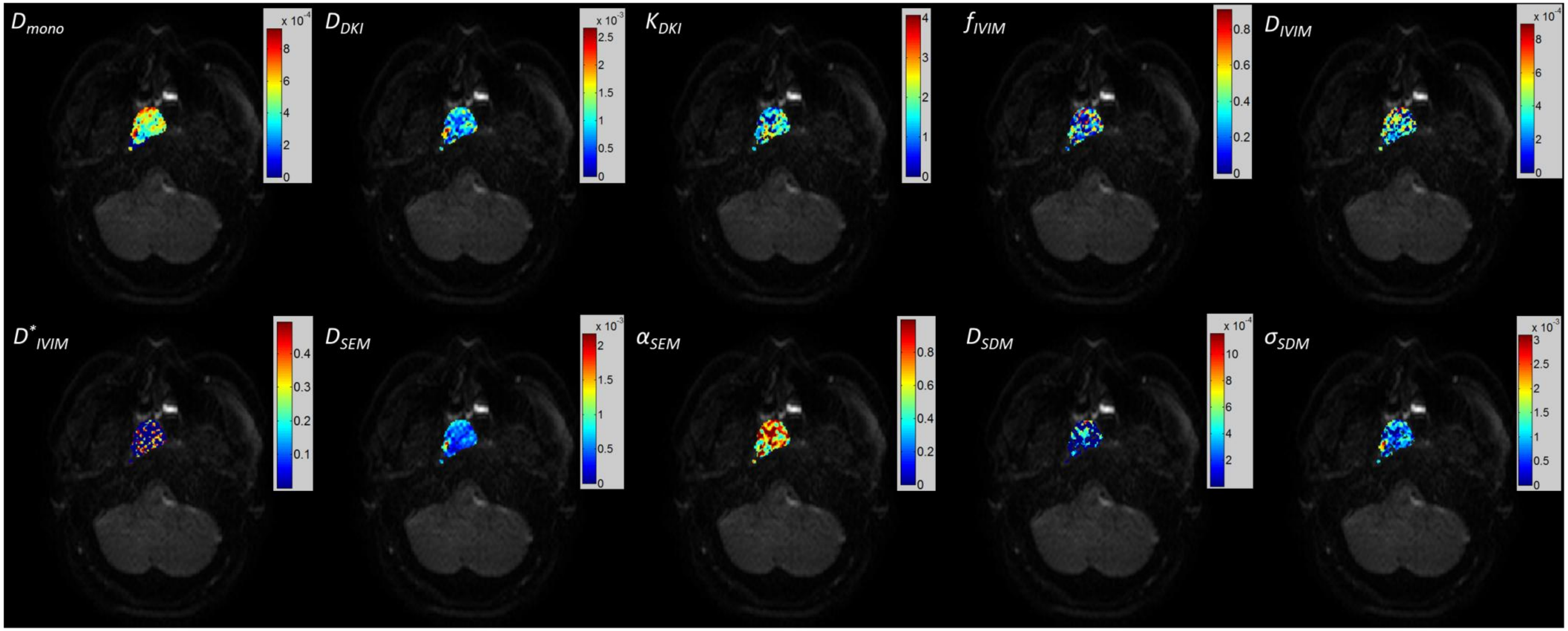
128x128 pixel image
x
number of slice (~10-40)
x
number of diffusion gradient directions (≥15)

Typical non-Gaussian DW-NMR imaging post-processing



- Cumulant Expansion approach: Diffusional Kurtosis Imaging
- Stretched Exponential model ImagingState of the art of non-Gaussian DW-NMR imaging

Why using non-Gaussian DW-NMR imaging post-processing

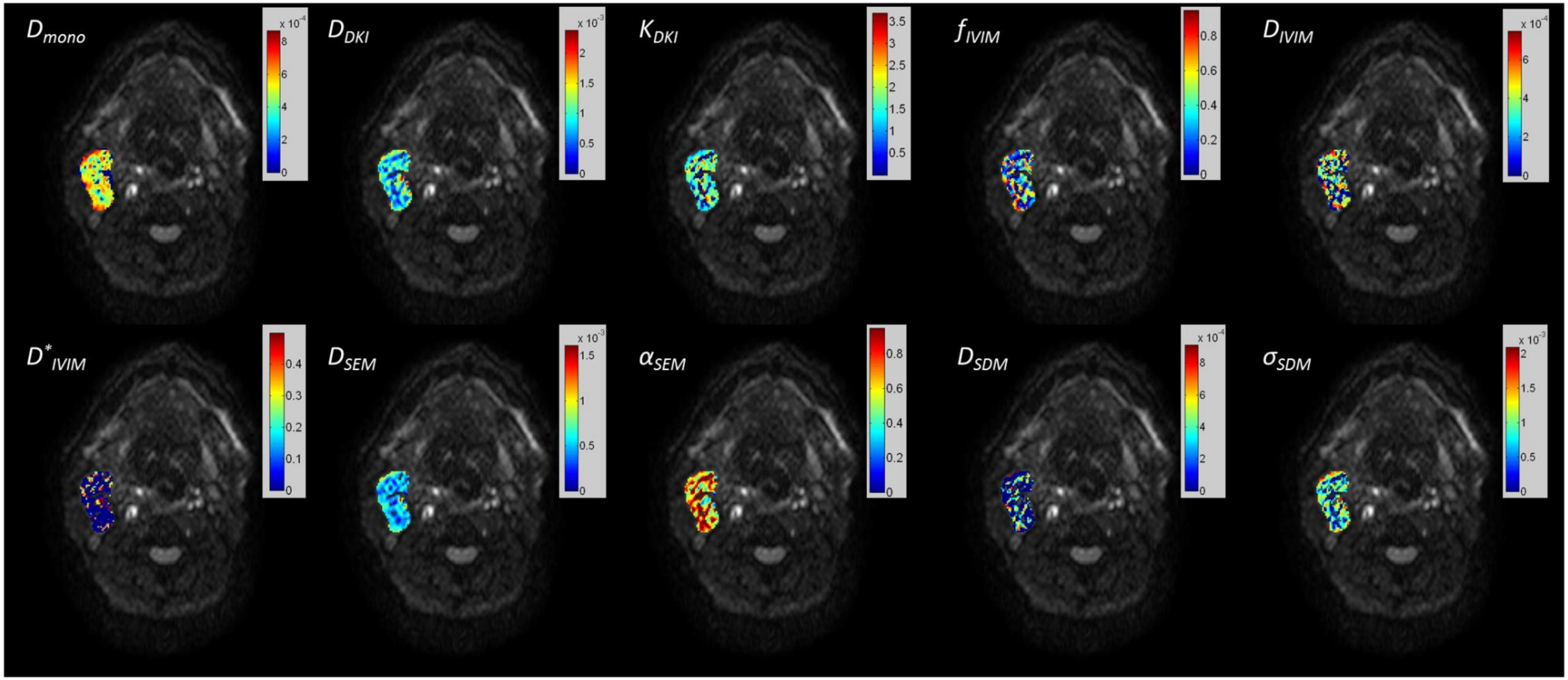


Nasopharyngeal Carcinoma

Maps of Gaussian and non-Gaussian parameters for the primary tumor

Yuan, Jing, et al. PloS one 2014 9(1); e87024.

Why using non-Gaussian DW-NMR imaging post-processing

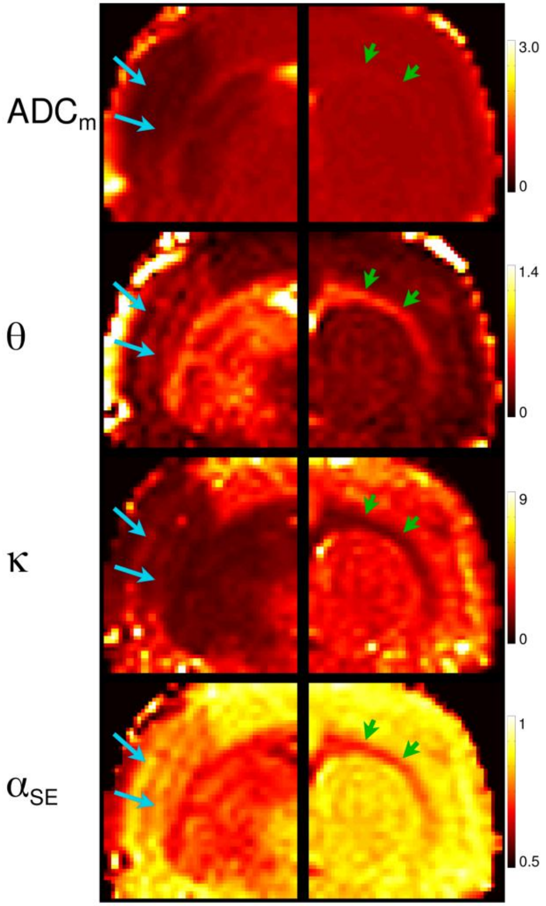


Nasopharyngeal Carcinoma

Maps of Gaussian and non-Gaussian parameters for the metastatic node

Yuan, Jing, et al. PloS one 2014 9(1); e87024.

Why using non-Gaussian DW-NMR imaging post-processing



Ischemic Stroke

Non Gaussian diffusion maps best representing the **ischemic lesion** in each of the three animals.

Non-Gaussian parameters **best represent the layered structure** in the lesions and **WM tracts**

Grinberg, Farida, et al. PloS one 2014 9(2); e89225.

Why using non-Gaussian DW-NMR imaging post-processing

Non-Gaussian DW-NMR imaging increases the specificity and sensitivity of DW-NMR techniques in determining pathological tissue (in both diagnosis and surgery phases)

Limitations:

The non-Gaussian processing is not currently available in "real-time" mode (in a few seconds)

DIFFUSIONAL KURTOSIS IMAGING

$$S(b) \propto e^{-D_{app} b + \frac{1}{6} K_{app} \left[D_{app} b\right]^2 + \dots}$$

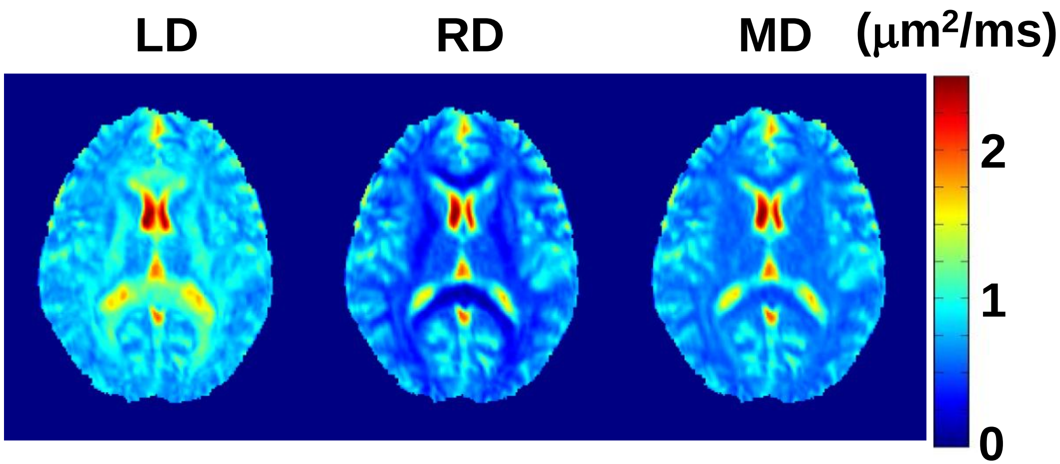
$$S(b) = S(0)\exp\left[-b\sum_{i,j} n_i n_j D_{ij} + \frac{1}{6} b^2 \left(\frac{1}{3} \sum_i D_{ii} \right)^2 \sum_{i,j,k,l} n_i n_j n_k n_l W_{ijkl} + O(b^3) \right]$$

N ≥ 15 independent measures to get W_{ijkl}

$$K(\mathbf{n}) = \frac{\overline{D}^2}{[D(\mathbf{n})]^2} \sum_{i,j,k,l=1}^{3} n_i n_j n_k n_l W_{ijkl}.$$

$$\boxed{MK = \frac{1}{N} \sum_{i=1}^{N} K(\mathbf{n}_i)}$$

DIFFUSIONAL KURTOSIS IMAGING



Conventional DTI

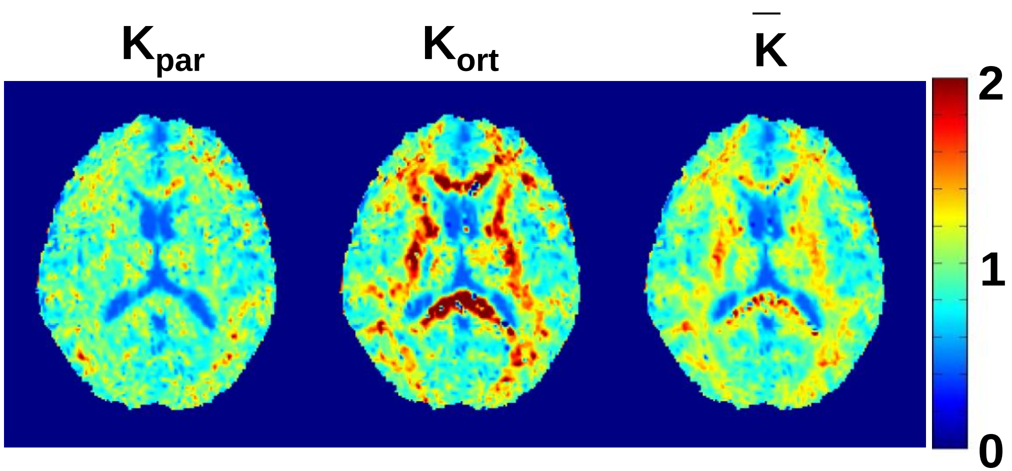
128x128x32x15

=

7.864320 10^6

linear systems to solve

~ 15 s on CPU*



Kurtosis tensor imaging

128x128x32x15

=

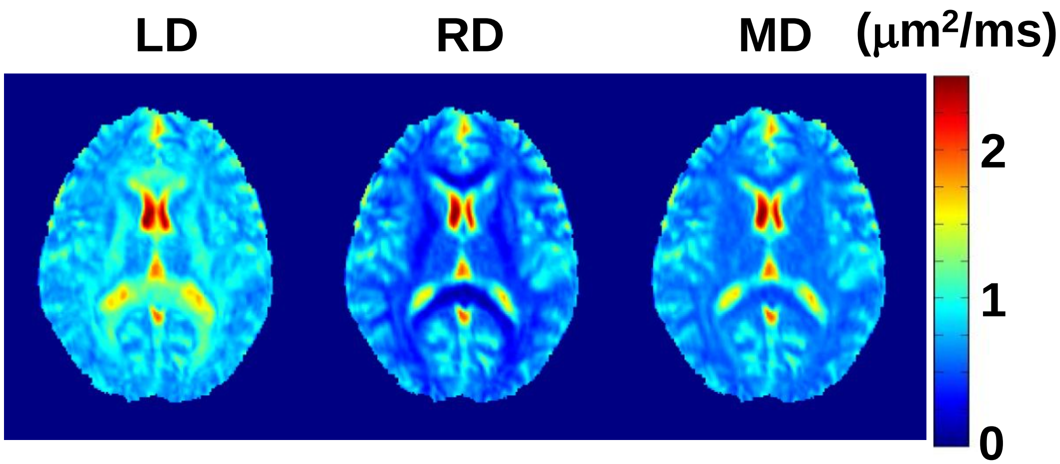
7.864320 10^6

non-linear fit to perform

~7200 s on CPU*

* 8 threads on an Intel Xeon E5-2609 CPU at 2.4 GHz

DIFFUSIONAL KURTOSIS IMAGING



Conventional DTI

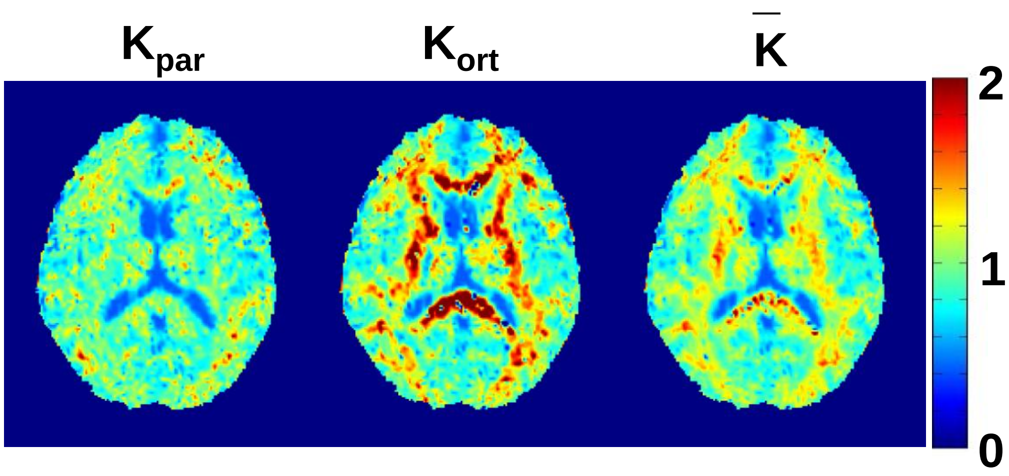
128x128x32x15

=

7.864320 10^6

linear systems to solve

~ 15 s on CPU*



Kurtosis tensor imaging

128x128x32x15

=

7.864320 10^6

non-linear fit to perform

~7200 s on CPU*

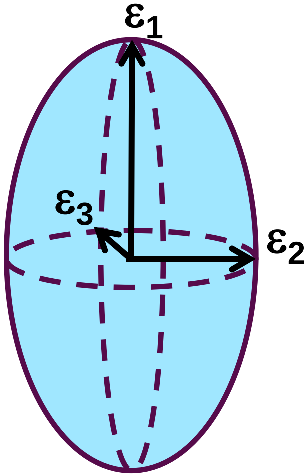
* 8 threads on an Intel Xeon E5-2609 CPU at 2.4 GHz

$$S(b) = S(0)e^{-A_1(b_1^*)^{\gamma_1} - A_2(b_2^*)^{\gamma_2} - A_3(b_3^*)^{\gamma_3}}$$

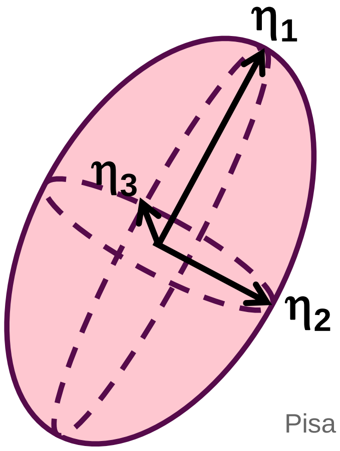
$$b_i^* = \vec{b} \cdot \vec{\eta}_i$$

Anomalous diffusion reference frame

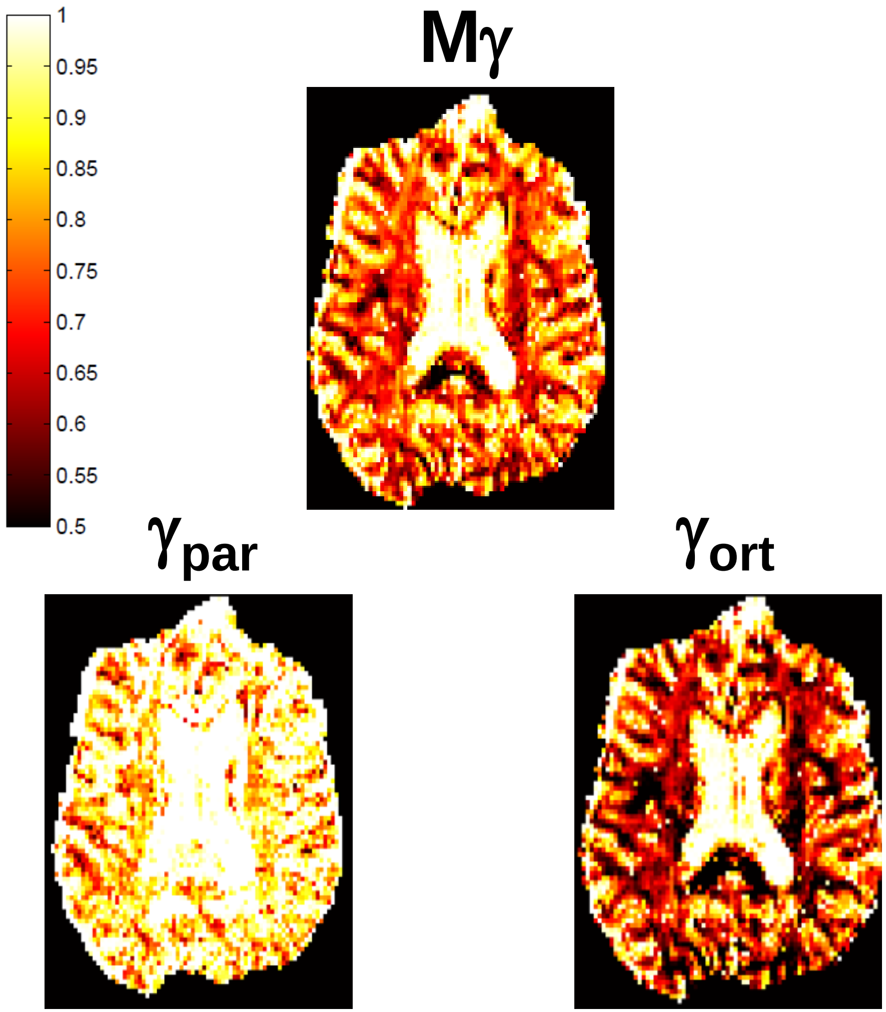
DTI reference frame



Anomalous Diffusion reference frame



STRETCHED EXPONENTIAL MODEL IMAGING



Conventional DTI

128x128x32x15

=

7.864320 10^6

linear systems to solve

~ 15 s on CPU*

Stretched exponential imaging

128x128x32x15

=

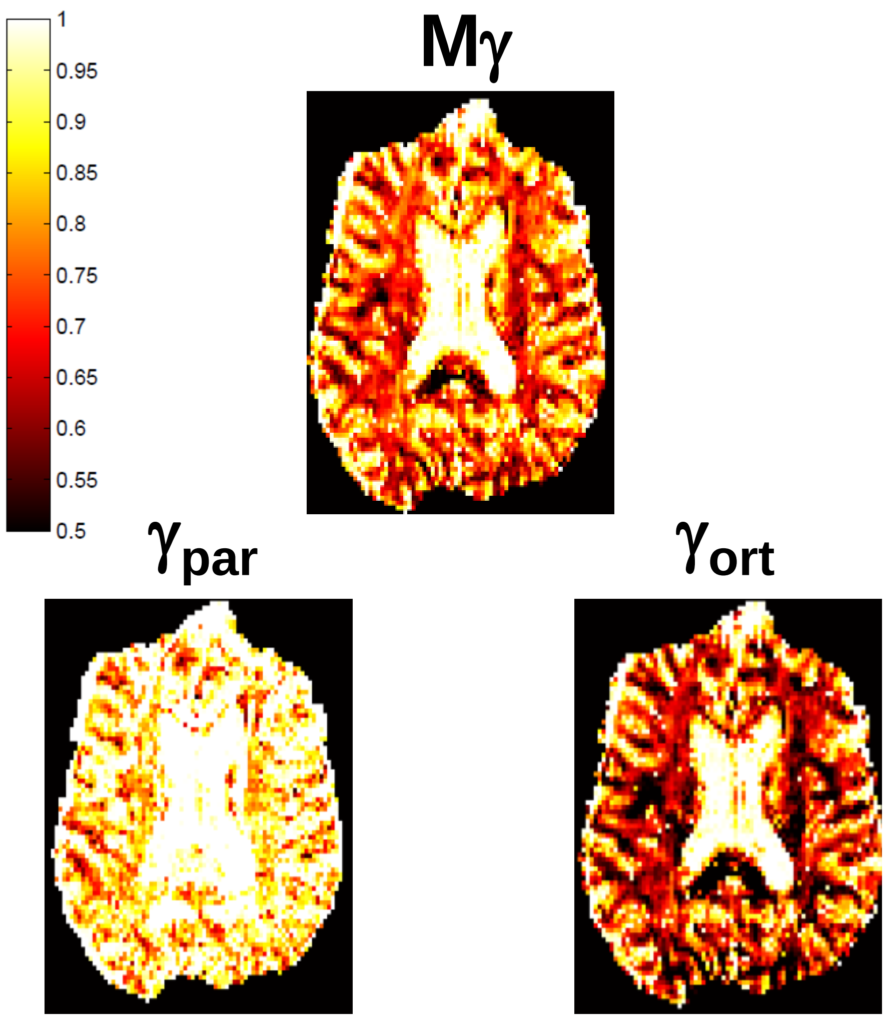
7.864320 10^6

non-linear fit to perform

~6600 s on CPU*

* 8 threads on an Intel Xeon E5-2609 CPU at 2.4 GHz

STRETCHED EXPONENTIAL MODEL IMAGING



Conventional DTI

128x128x32x15

=

7.864320 10^6

linear systems to solve

~ 15 s on CPU*

Stretched exponential imaging

128x128x32x15

=

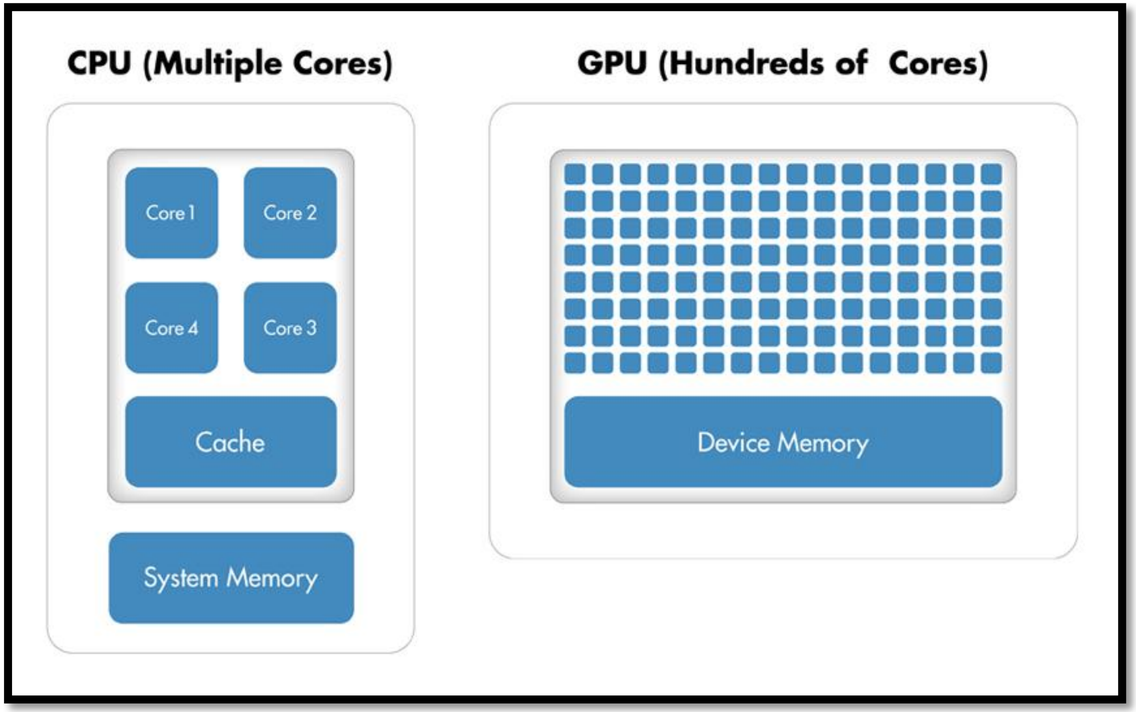
7.864320 10^6

non-linear fit to perform

~6600 s on CPU*

* 8 threads on an Intel Xeon E5-2609 CPU at 2.4 GHz

WHY AND HOW USING GPU IN DW-NMR IMAGING



- **Computationally intensive** — The time spent on computation significantly exceeds the time spent on transferring data to and from GPU memory.
- **Massively parallel** — The computations can be broken down into hundreds or thousands of independent units of work.

Computationally intensive

Kurtosis tensor imaging
non-linear fit
~7 ms per pixel
vs
~ 10-80 ns to read data from global memory

Stretched exponential imaging
non-linear fit
~6 ms per pixel
vs
~10-80 ns to read data from global memory

Massively parallel

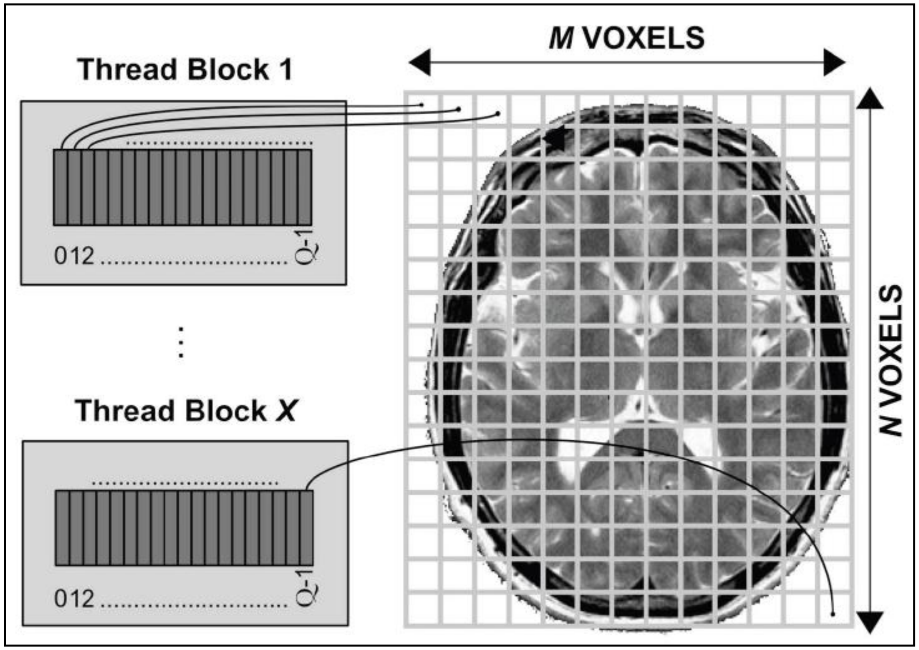
Kurtosis tensor imaging
~10^6-10^7 **independent** non-linear fit

Stretched exponential imaging
10^6-10^7 **independent** non-linear fit

- The **Levenberg-Marquardt** algorithm is based on an iterative numerical optimization procedure that minimizes the sum of squared model residuals.

- The CUDA kernel performs the Levenberg-Marquardt algorithm. This kernel maps each CUDA thread to a voxel, and it launches as many threads as voxels contained in a particular slice [1,2].

- Because processing of different voxels is totally independent, the threads do not need to synchronize.



[1] Zhu, X., Zhang, D. *PloSone* 2013, 8(10),e76665

[2] Hernández, M. et al. *PloSone* 2013 8 (4),e61892.

WHY AND HOW USING GPU IN DW-NMR IMAGING

Model function: $f(x_i,\xi)$

Experimental data: y_i

Tentative result: $P_i = \{\xi\}_i$

Initial parameters $\{\xi\}_i$

Compute the jacobian: $J=\delta f/\delta\xi$

Compute the error gradient: $G=J^T(y_i-f(x_i,\xi))$

Compute the cross-product jacobian: $H=J^TJ$

Solve $(H+\lambda \, diag(H))\mu=G$ Obtain μ

$\xi = \xi + \mu$

$<(y_i-f(x_i,\xi))^2>$ Satisfy convergence criteria

yes

STOP and take ξ

no

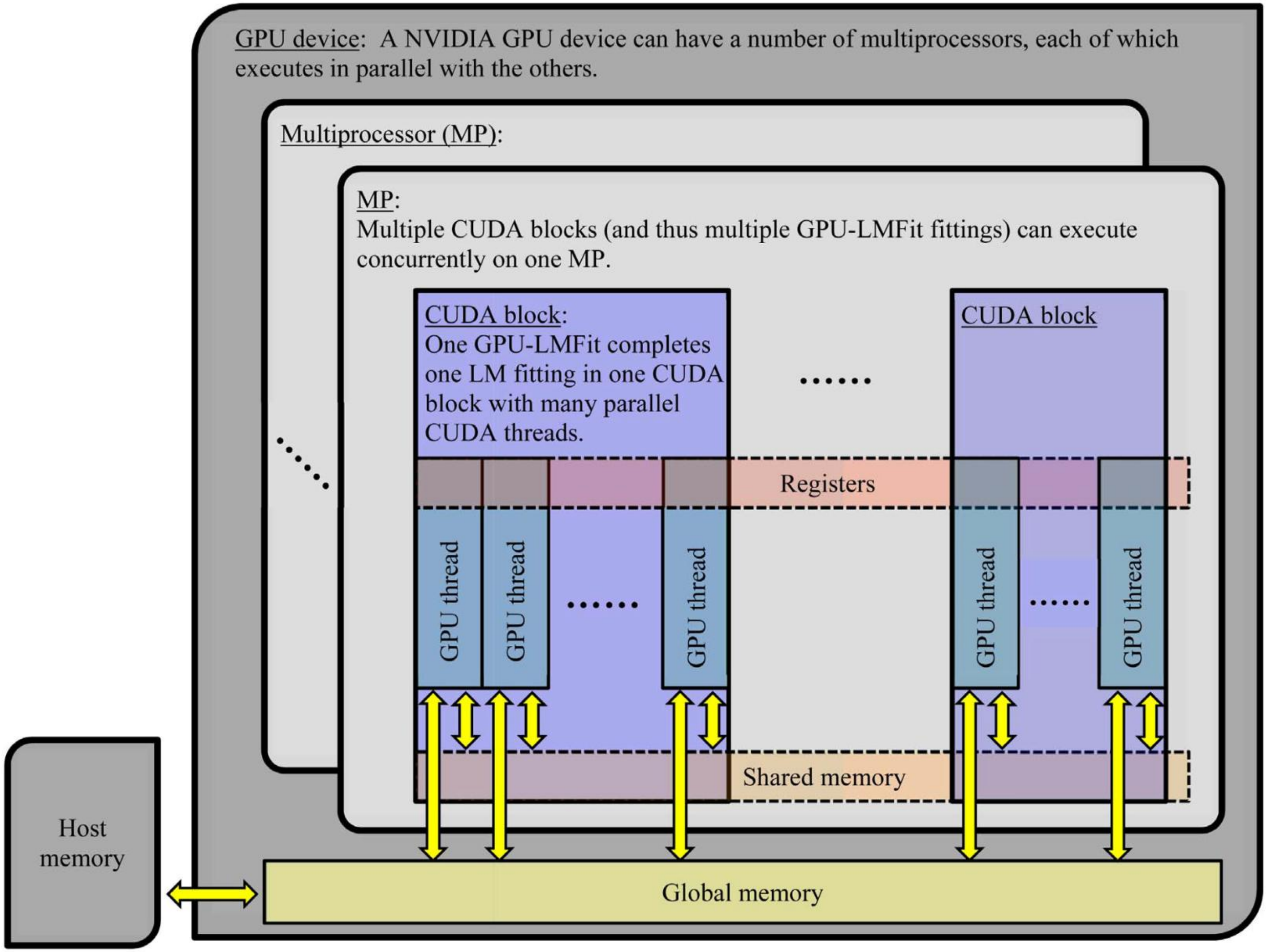
reject

If $<(y_i-f(x_i,\xi))^2> \uparrow$ $\lambda \uparrow$

accept

If $<(y_i-f(x_i,\xi))^2> \downarrow$ $\lambda \downarrow$

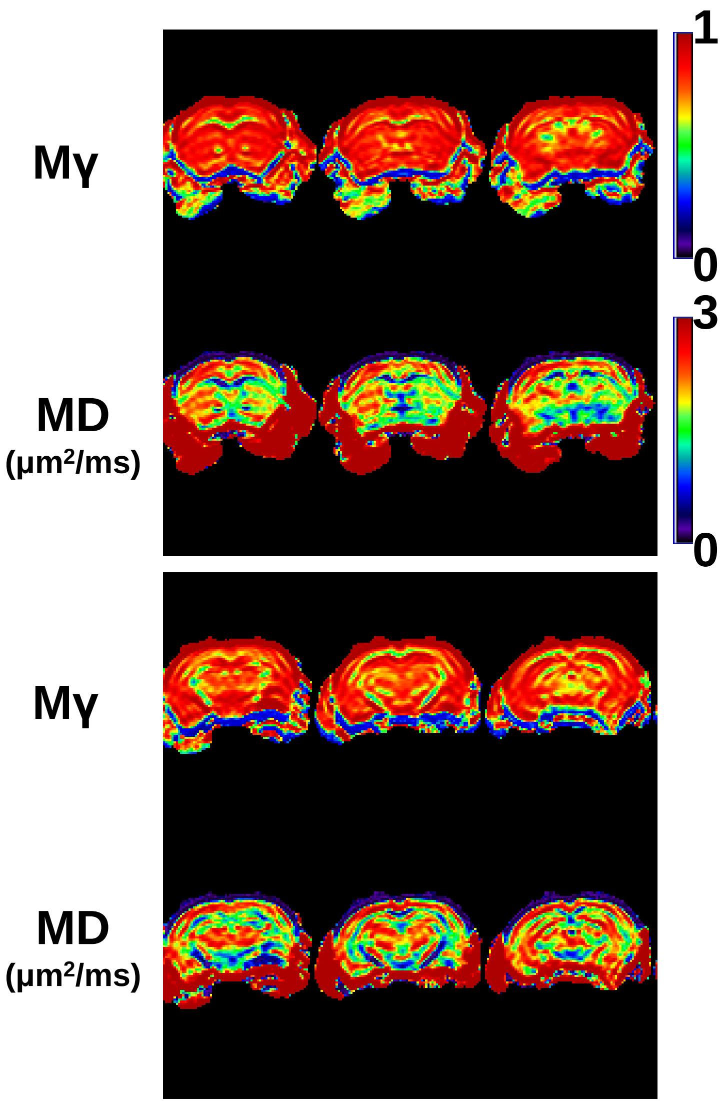
WHY AND HOW USING GPU IN DW-NMR IMAGING



[1] Zhu, X., Zhang, D. *PloSone* 2013, 8(10),e76665

APPLICATION: NON-GAUSSIAN DW-NMR IMAGING OF MOUSE BRAIN ON GPU

GPU optimized Levenberg-Marquardt algorithm for non-linear fitting.

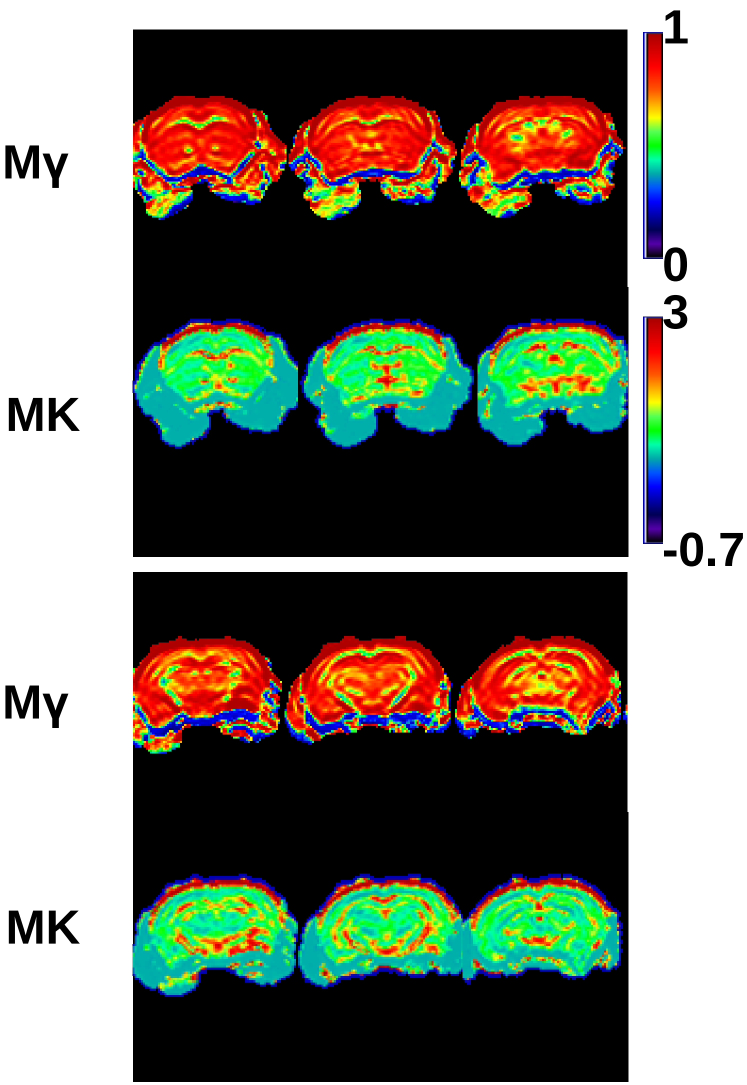


A **Nvidia Quadro K2000** with **2Gb** of **dedicated memory**, which supports **1024 threads per block**, with a **maximum number of 64 registers per thread**, was used for the analysis

DW-NMR data-set	≈ 121.521 Mb ≈ $5x10^5$ voxels
CUDA blocks	8
CUDA Grid	(4096, 1, 1)
Total used global memory on GPU	≈ 7 – 15 Mb
Average Speed	≈ 12000 fit/s GPU ≈ 50 fit/s CPU
Computing time	≈ 40 s GPU ≈ 9720 s CPU
Speed-up factor	≈ 240

FUTURE PERSPECTIVES: DW-NMR IMAGING

GPU optimized Parallel Tempering algorithm for non-linear fitting.



A **Nvidia Quadro K2000** with **2Gb** of **dedicated memory**, which supports **1024 threads per block**, with a **maximum number of 64 registers per thread**, was used for the analysis

DW-NMR data-set	≈ 121.521 Mb ≈ $5x10^5$ voxels
CUDA blocks	8
CUDA Grid	(4096, 1, 1)
Total used global memory on GPU	≈ 7 – 15 Mb
Average Speed	≈ 18000 fit/s GPU ≈ 50 fit/s CPU
Computing time	≈ 30 s GPU ≈ 9720 s CPU
Speed-up factor	≈ 360

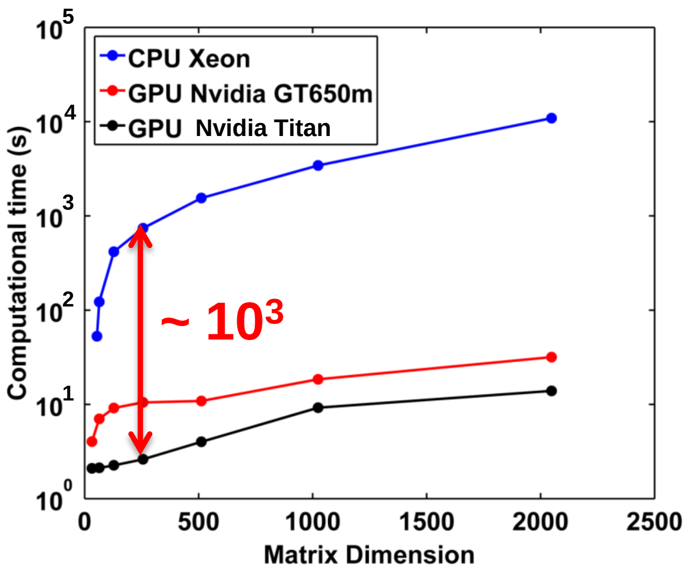
FUTURE PERSPECTIVES: DW-NMR IMAGING

Potentiality of GPU optimized algorithms for non-linear fitting.

Simulated expected speed-up

for non-Gaussian DW-NMR

Imaging from commercial and one of the best Nvidia GPUs available



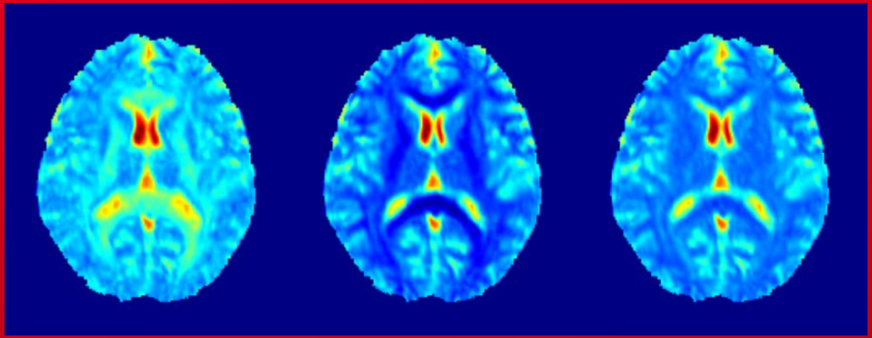
Potential speed-up thanks to one of the best Nvidia GPU available

DW-NMR data-set	≈ 121.521 Mb ≈ $5x10^5$ voxels
CUDA blocks	8
CUDA Grid	(4096, 1, 1)
Total used global memory on GPU	≈ 7 – 15 Mb
Average Speed	≈ fit/s GPU ≈ 50 fit/s CPU
Computing time	≈ 10 s GPU ≈ 9720 s CPU
Speed-up factor	≈ 1000

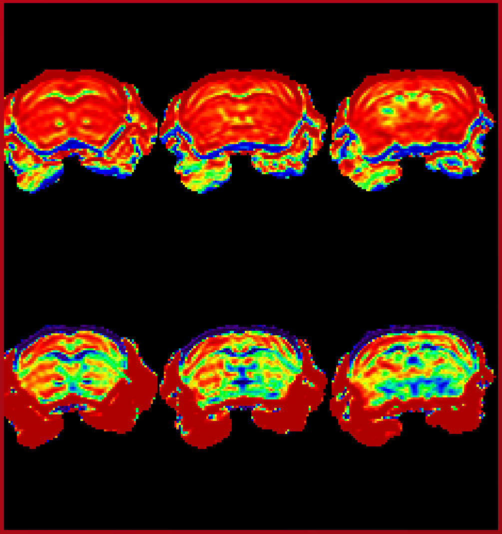
CONCLUSION

- A pixel-wise approach by using fast, accurate and robust parallel minimization optimizers, was implemented on GPU.
- A **dramatic speed-up** (**~10^2-10^3**) in massive non-linear model fitting analysis was obtained with respect to new generation multi-core processors.
- These results suggest that **real-time** automated pixel-wise parametric DW-NMR imaging is a promising application of GPUs.

Thank you for your attention



Special thanks to my co-authors



If interested, please contact me at:

marco.palombo@cea.fr