GPU accelerated de-dispersion for LOFAR transient searches

W. Armour, M. Giles, A. Karastergiou and C. Williams.
University of Oxford.

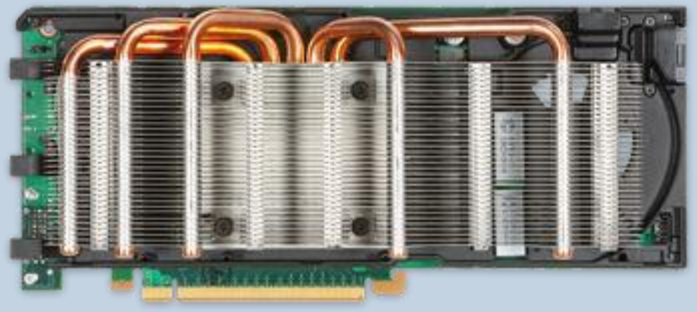
14th June 2012

Why use GPUs ?

Latest Kepler based cards

- GeForce GTX 690 – 2x GK104 GPUs
- 18.7 GFLOPS / watt
- Peak performance ~ 5.6 TFLOPS
- Price point ~ £1000

- K10 – 2x GK104 GPU 2x1536 cores.
- Peak performance 4.6 TFLOPS
- 20.4 GFLOPS / watt
- Price point ~ £3000 (?)



Influence and Take-up

TOP 500 – 3 out of top 5 utilise Fermi/Tesla

- Tianhe-1A 2.5 petaflops
- Based on 14336 Xeon and 7168 M2050
- To achieve same performance using only CPUs 50000 CPUs, 2x floor space and 3x power (Estimates made by NVIDIA)
- Uses Lustre :-s

The MOTIVATE project…

- MOTIVATE is a pathfinder project and aims to investigate the latest many-core technologies with the aim of delivering energy and cost efficiency in the area of radio astronomy HPC.

- MOTIVATE stands for Many-cOre Technology Investigating Value, Application, deploymenT and Efficiency.

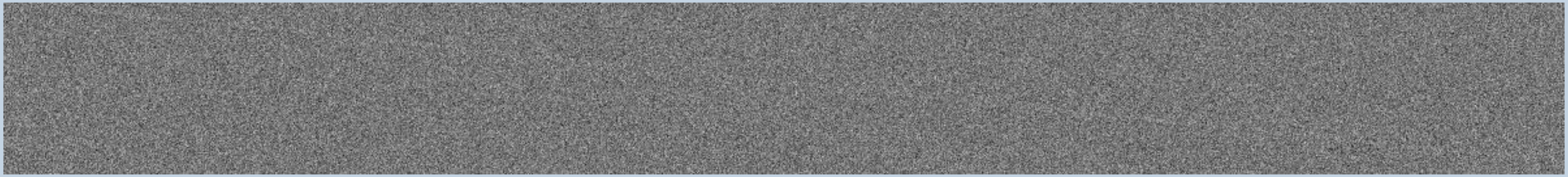
- The MOTIVATE project is funded by the Oxford-Martin School through the Institute for the Future of Computing

http://www.oerc.ox.ac.uk/research/many-core

De-dispersion

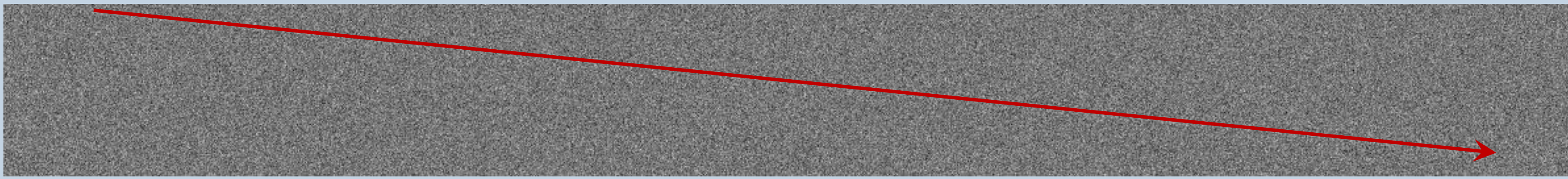
Experimental data

Most of the measured signals live in the noise of the apparatus.

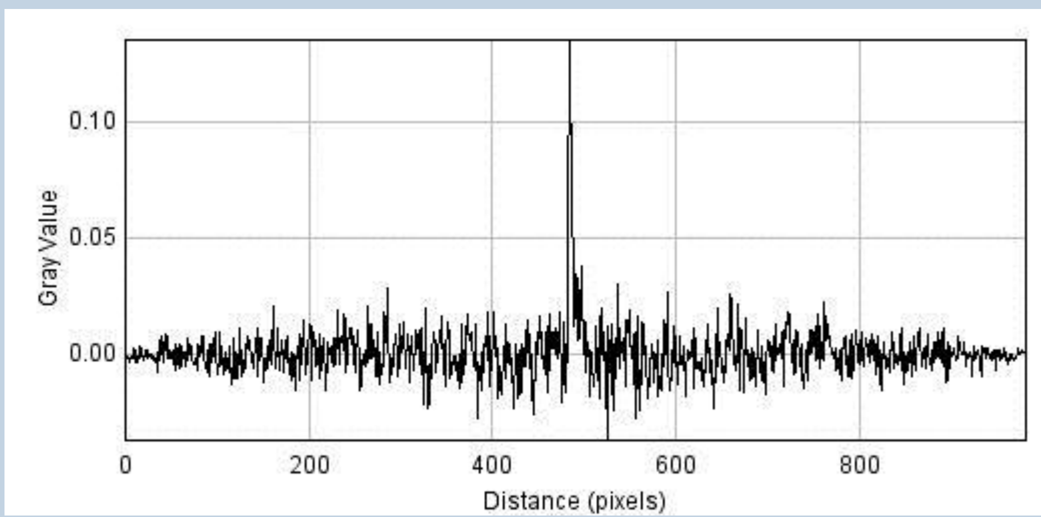


Experimental data

Most of the measured signals live in the noise of the apparatus.



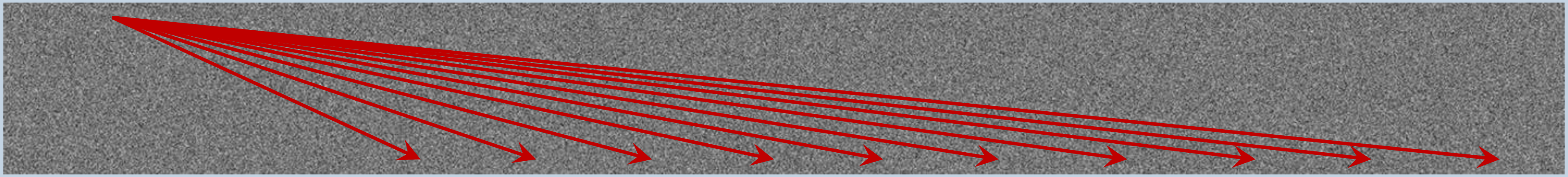
Hence frequency channels have to be “folded”



OXFORD e-Research CENTRE

Brute force algorithm

Every DM is calculated to see if a signal is present.



- In a blind search for a signal many different dispersion measures are calculated.
- This results in many data points in the (f,t) domain being used multiple times for different dispersion searches.
- This allows for data reuse in a GPU algorithm.

OXFORD e-Research CENTRE

A new GPU brute force algorithm

Three key features…

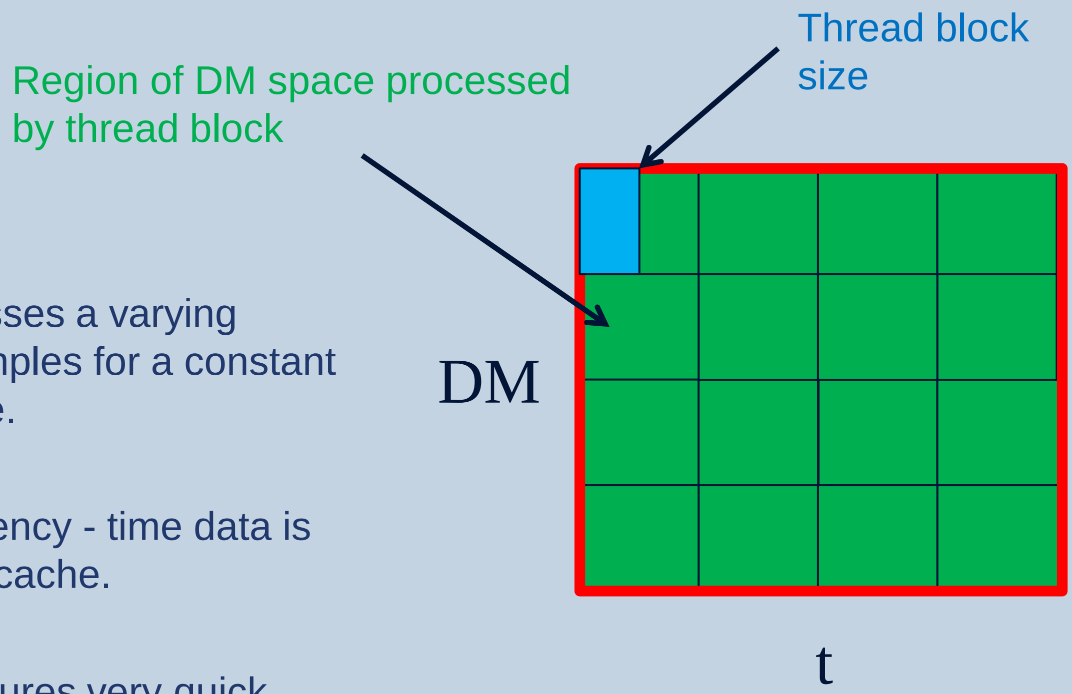
- Each thread processes a variable number of dispersion measures in local registers.
- Exploit the L1 Cache / Shared Memory present on the Fermi/Kepler architecture.
- Optimise the region of dispersion space being processed (thread blocksize).

Web page : http://www.oerc.ox.ac.uk/research/wes

OXFORD e-Research CENTRE

Processing several DM's per thread

New Algorithm works in the DM - t space rather than frequency – time space.

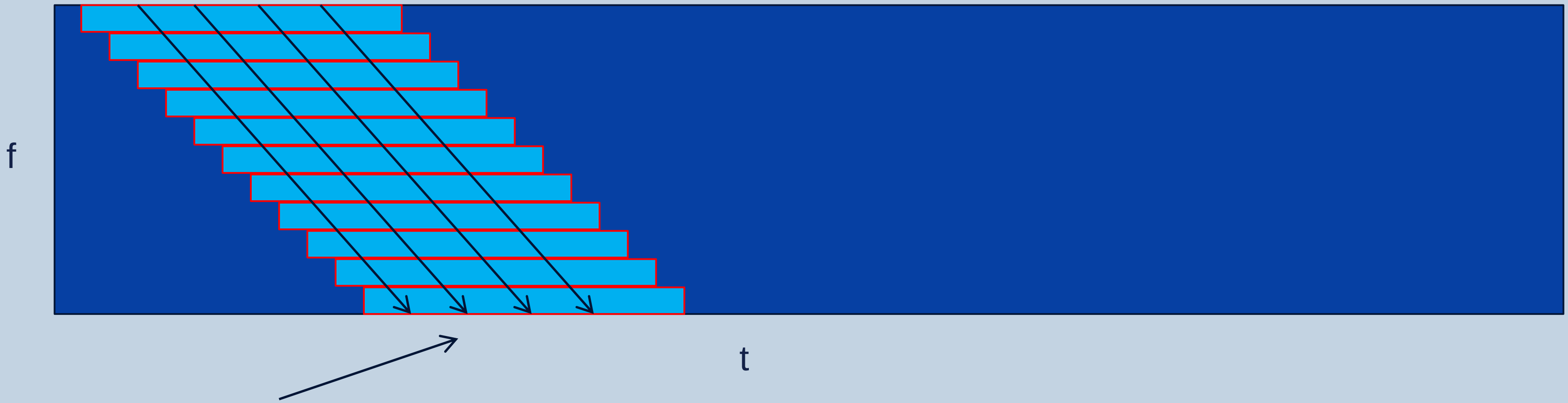


- Each thread processes a varying number of time samples for a constant dispersion measure.
- This ensures frequency - time data is loaded into fast L1 cache.
- Using registers ensures very quick memory access.

OXFORD e-Research CENTRE

Exploiting the L1 cache / Shared Memory…

Each dispersion measure for a given frequency channel needs a shifted time value.



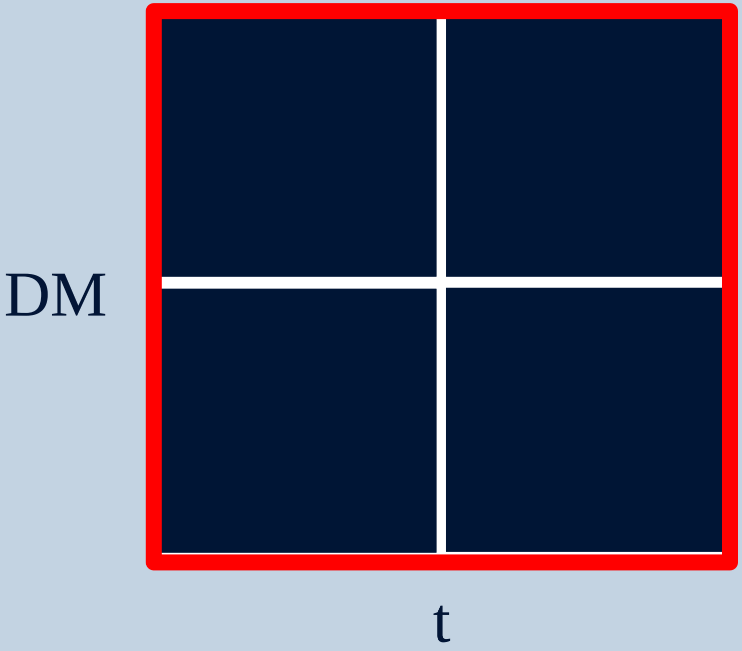
Constant DM's with varying time.

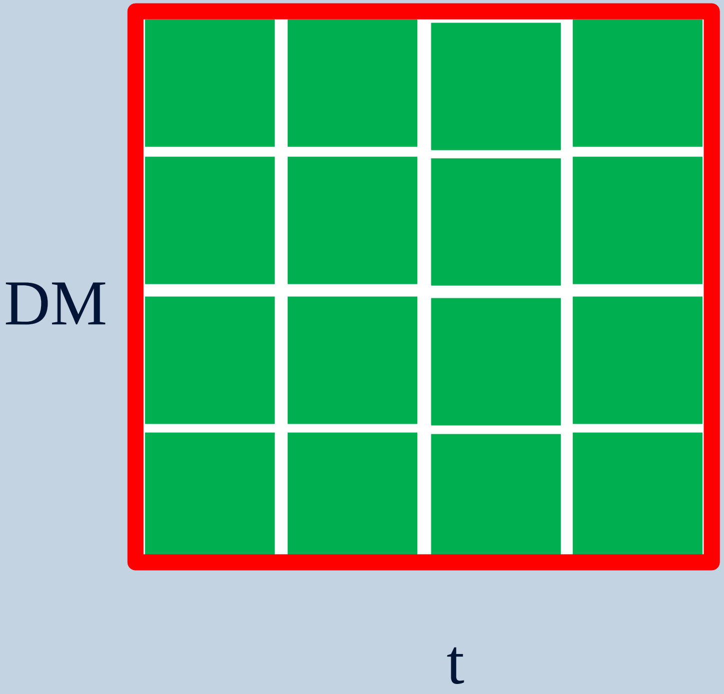
Incrementing all of the registers at every frequency step ensures a high data reuse of the stored frequency time data in the L1 cache.

OXFORD e-Research CENTRE

Optimising the parameterisation.

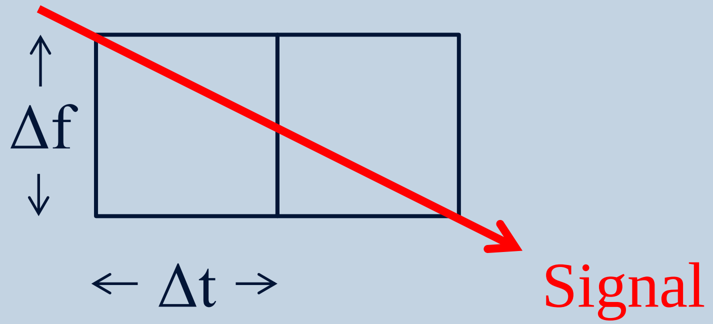
The GPU block size of the new algorithm can take on any size that is integer multiples of the size of a "data chunk"…



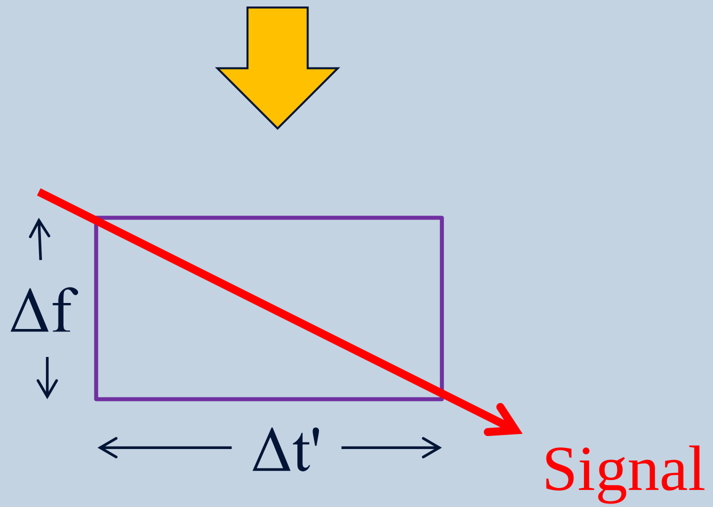


OXFORD e-Research CENTRE

Time binning



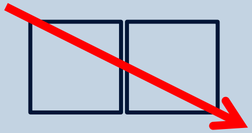
When scattering and dispersion effects are high a radio signal can be spread over multiple time samples, all having the same frequency.



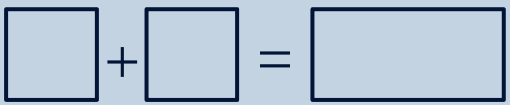
It makes sense to add the values of adjacent time data to increase the signal to noise. This reduces the amount of time samples to process at higher DMs (also increases the step size between DMs to achieve critical sampling)

Time binning…

1. Launch kernel to dedisperse data up to



2. Use GPU to bin data (in time) while concurrently transferring the de-dispersed data and analyze it on the host (CPU).



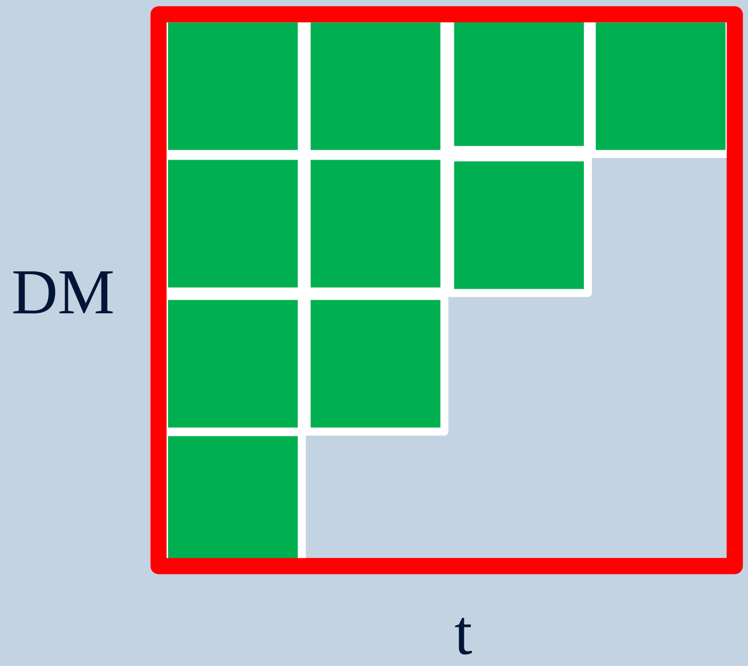
3. Use cudaThreadSynchronize() on host to ensure kernel execution has finished.

4. Repeat

Time binning…

Has the effect of reducing the amount threads that are needed to process a region of (DM,t) space.

Utilizes the CPU and GPU at the same time.



A CPU brute force algorithm

Four key features…

- Aim to achieve full cache line utilization.
- Exploit the large (~375 GB/s) LLC bandwidth present on the new Intel Sand Bridge CPUs.
- Use the Intel Intrinsics to exploit the 16 AVX/SSE (YMM/XMM) SIMD registers (don't rely on the Intel auto-vectorizer!)
- Use OpenMP to share work across the CPU cores.

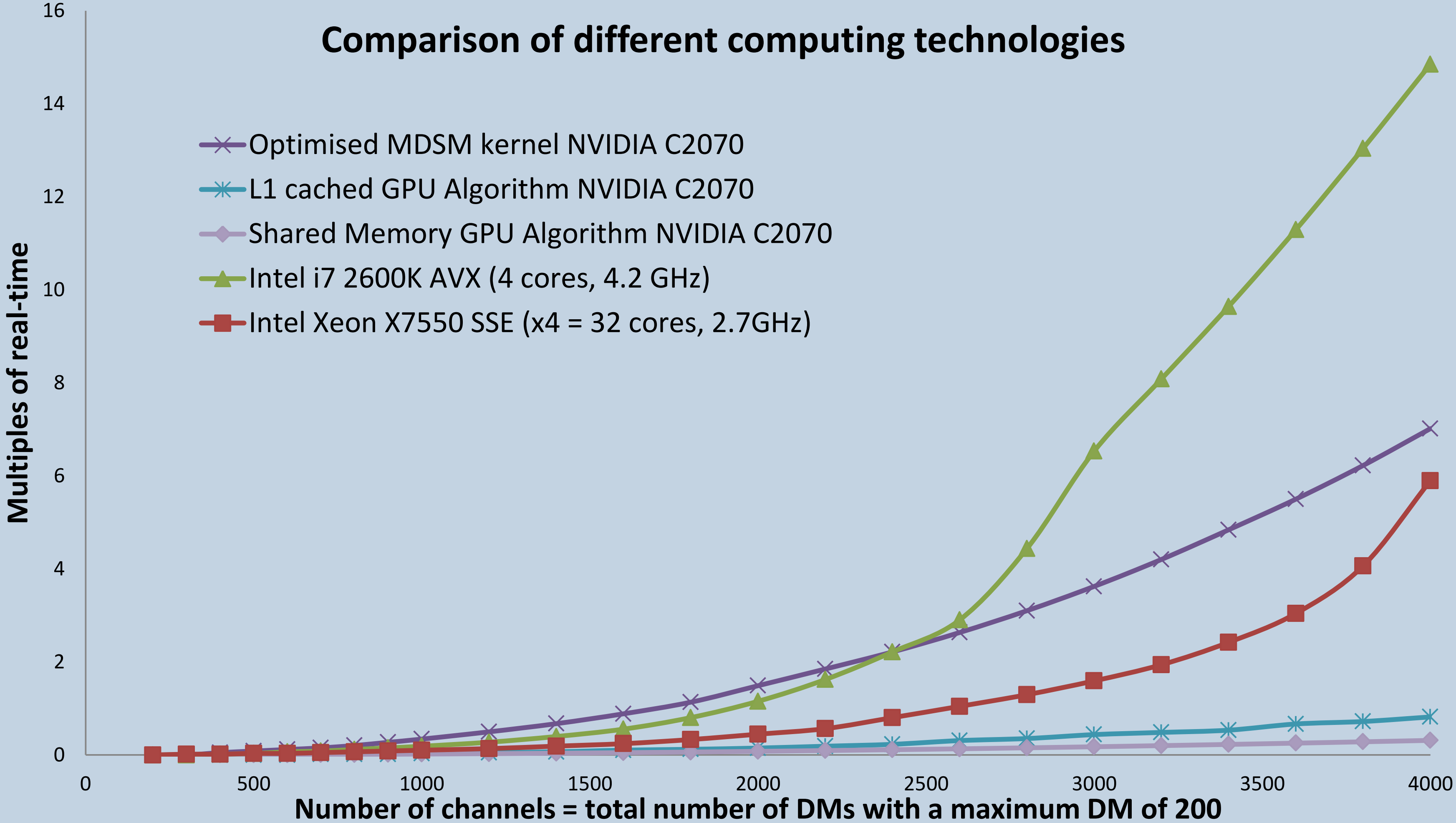
Intel experts can't accelerate this code any more than we have already

OXFORD e-Research CENTRE

Brute force results…

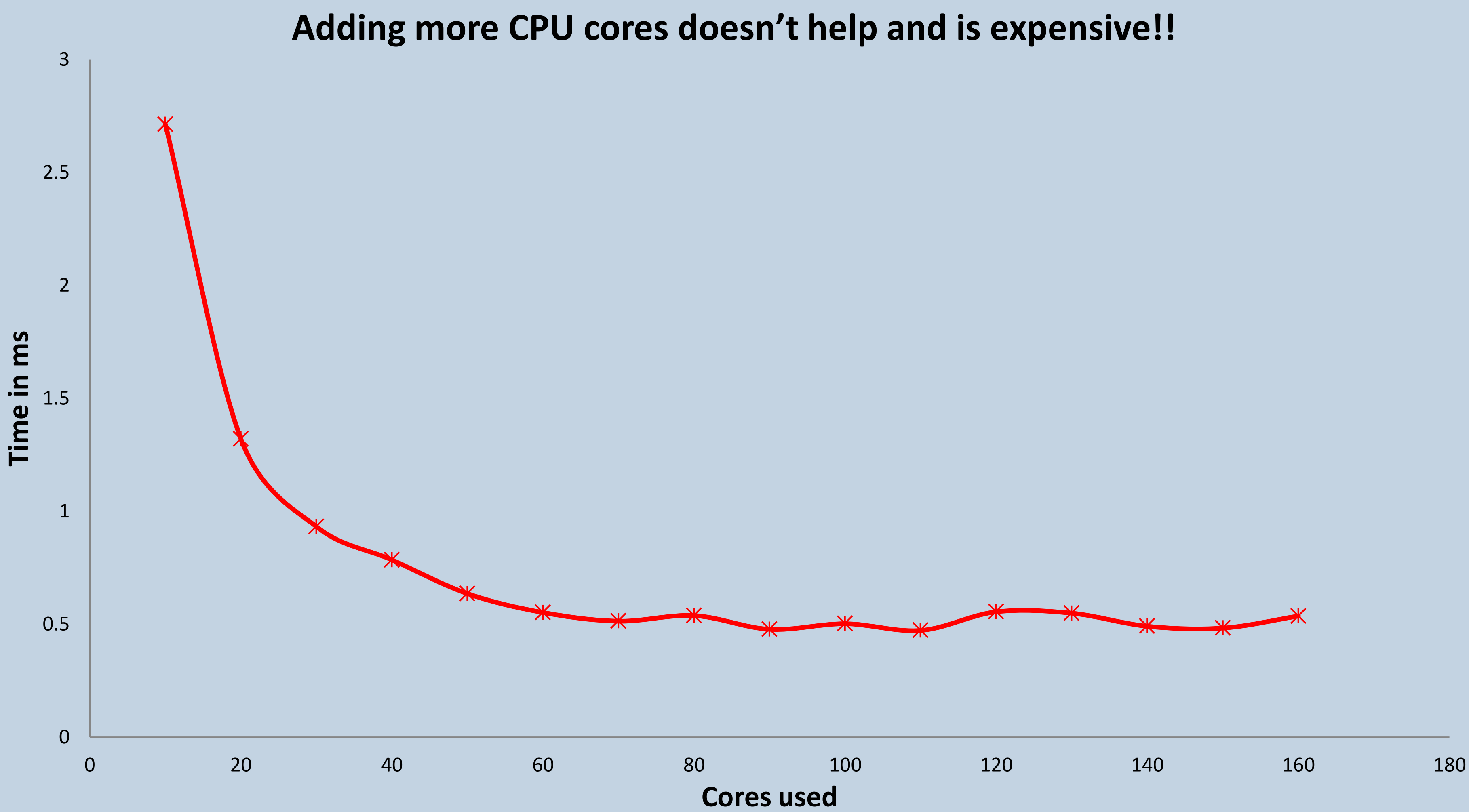
Results for LOFAR data

Results…

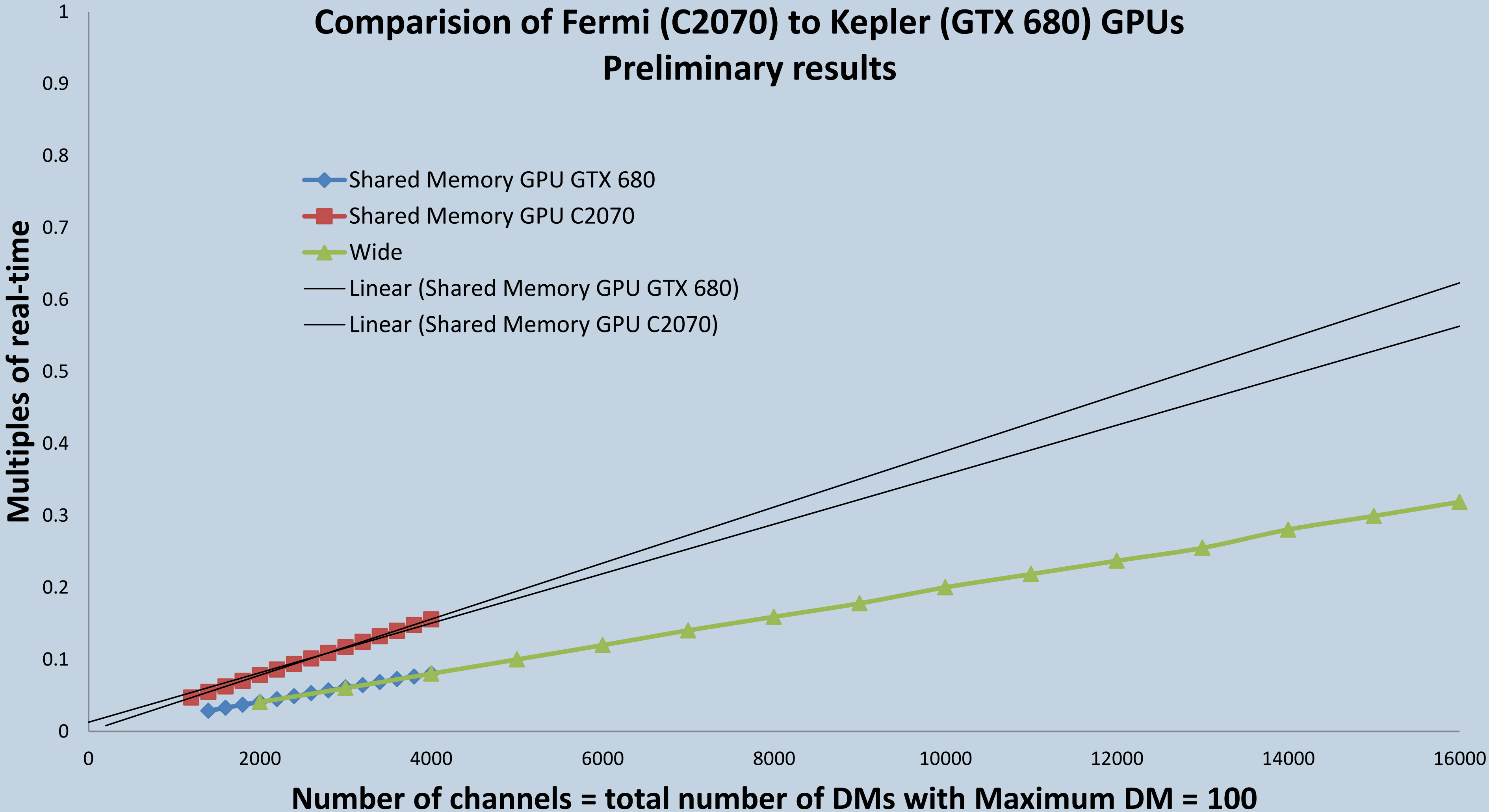


Comparison of different computing technologies
Optimised MDSM kernel NVIDIA C2070
L1 cached GPU Algorithm NVIDIA C2070
Shared Memory GPU Algorithm NVIDIA C2070
Intel i7 2600K AVX (4 cores, 4.2 GHz)
Intel Xeon X7550 SSE (x4 = 32 cores, 2.7GHz)
Multiples of real-time
0
2
4
6
8
10
12
14
16
0
500
1000
1500
2000
2500
3000
3500
4000
Number of channels = total number of DMs with a maximum DM of 200

Results…

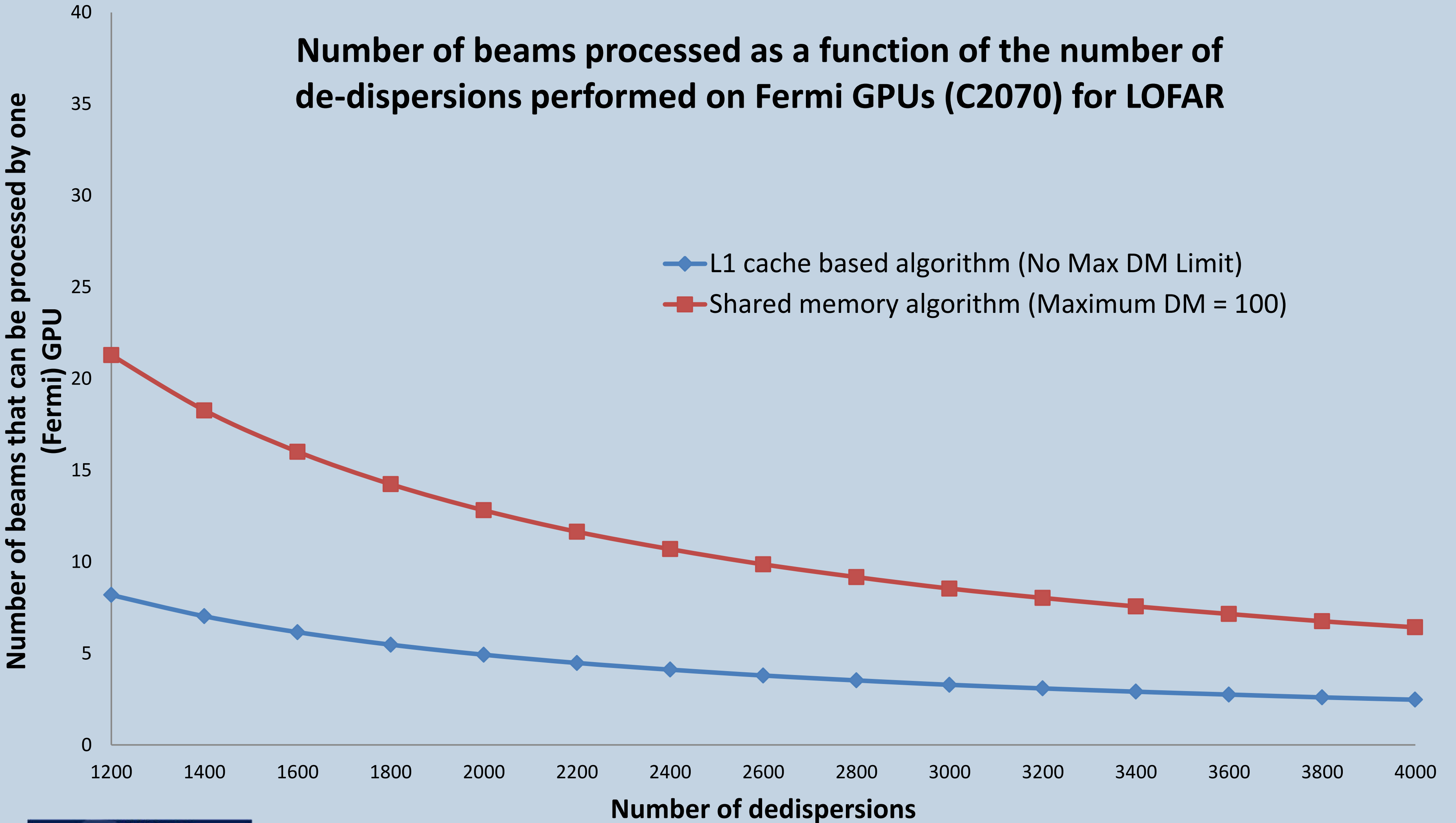

Adding more CPU cores doesn't help and is expensive!!
Time in ms
3
2.5
2
1.5
1
0.5
0
0
20
40
60
80
100
120
140
160
180
Cores used

Results…



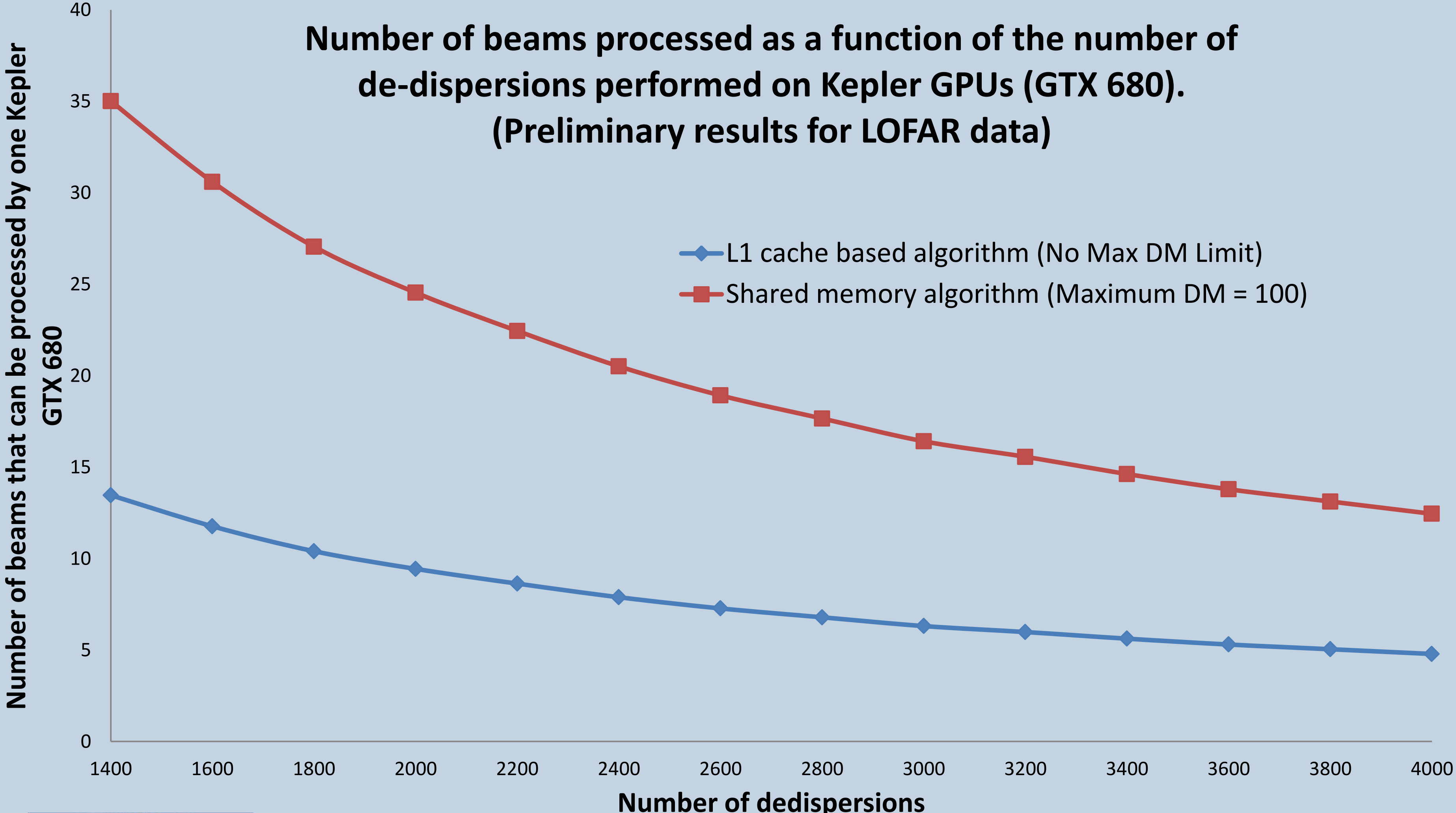
OXFORD e-Research CENTRE

Results…



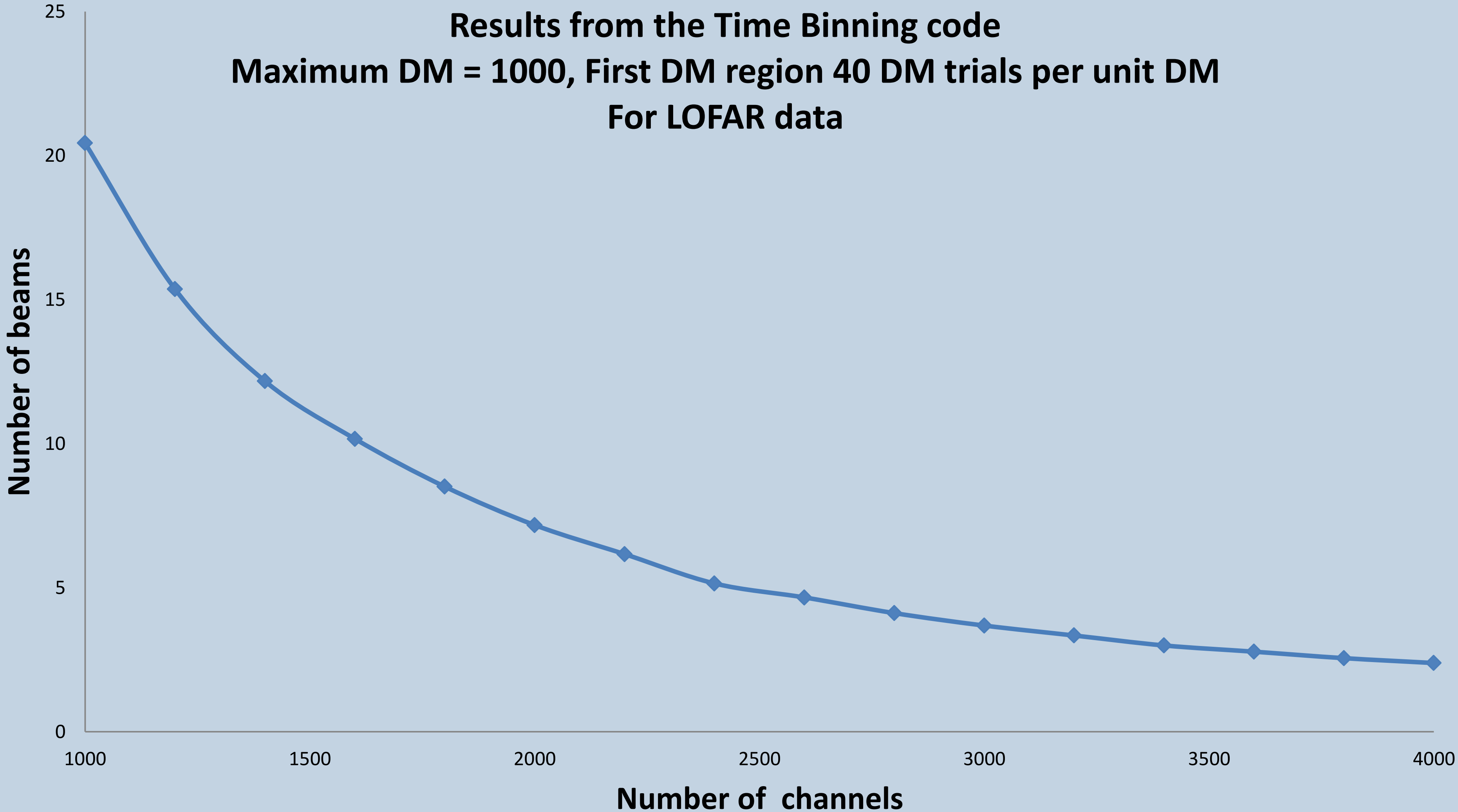
Number of beams processed as a function of the number of de-dispersions performed on Fermi GPUs (C2070) for LOFAR
L1 cache based algorithm (No Max DM Limit)
Shared memory algorithm (Maximum DM = 100)
Number of beams that can be processed by one (Fermi) GPU
Number of dedispersions
40
35
30
25
20
15
10
5
0
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
3200
3400
3600
3800
4000

Results…



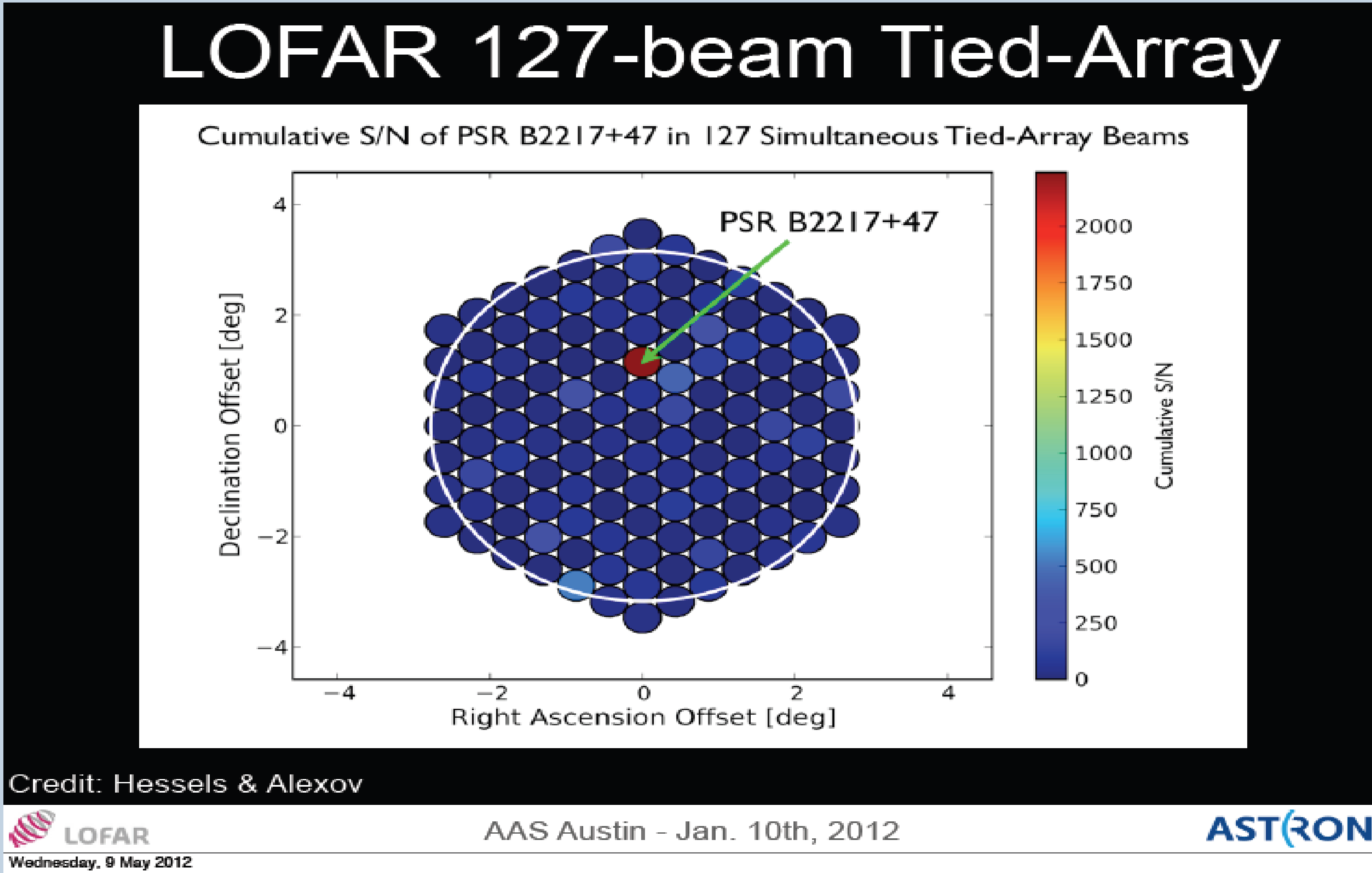
OXFORD e-Research CENTRE

Results…

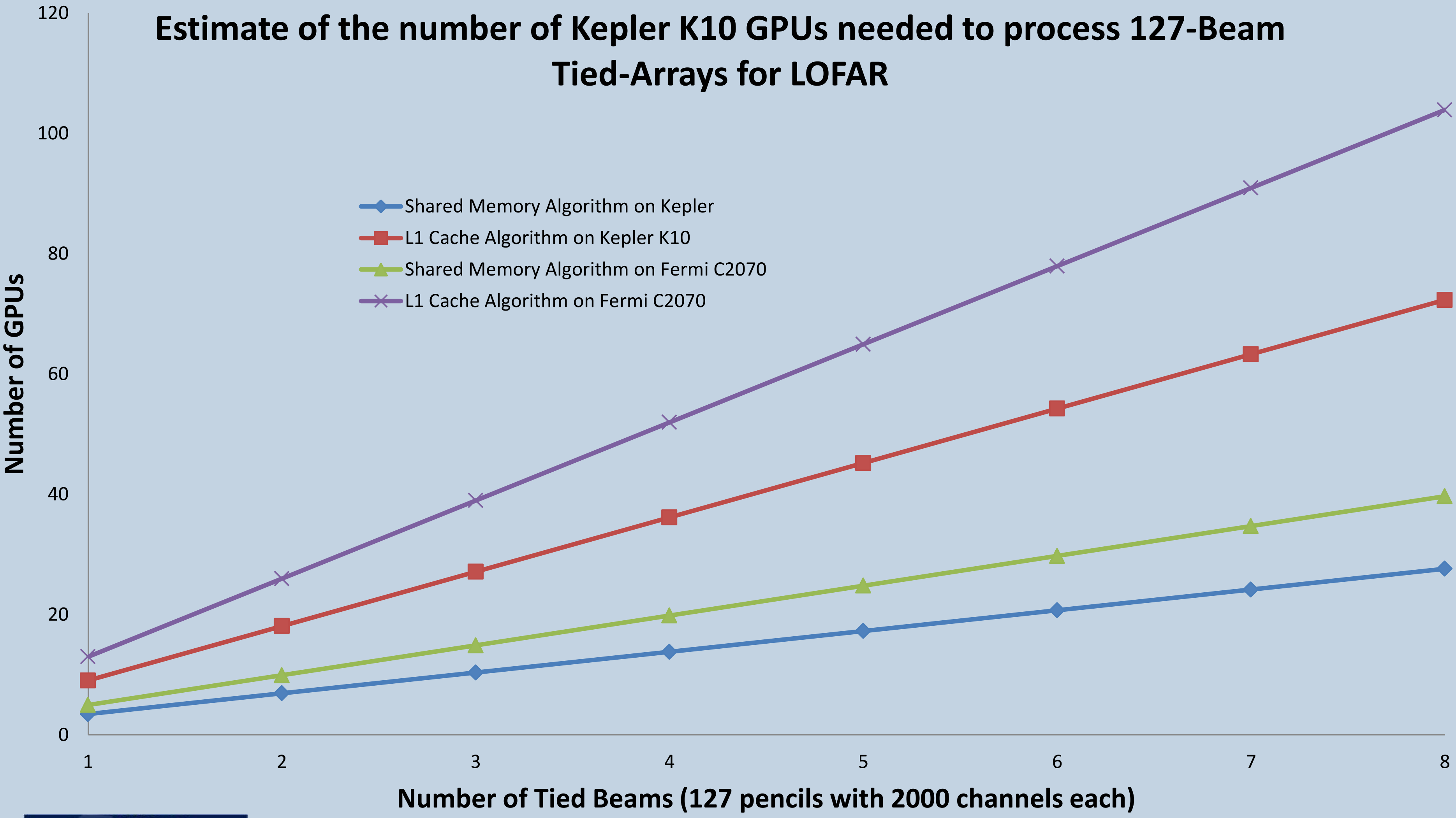


Results from the Time Binning code
Maximum DM = 1000, First DM region 40 DM trials per unit DM
For LOFAR data
Number of beams
Number of channels
0
5
10
15
20
25
1000
1500
2000
2500
3000
3500
4000

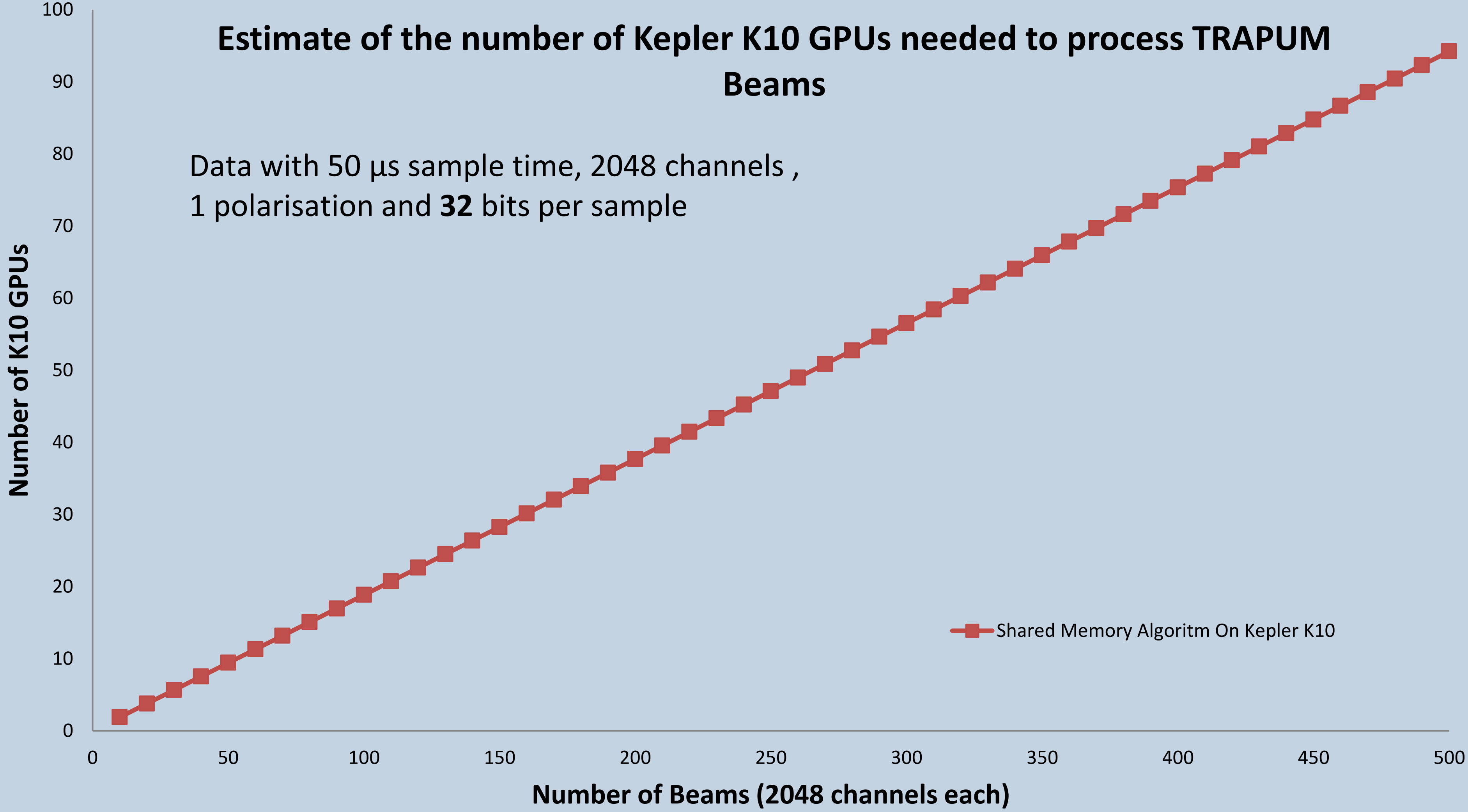
Conclusions and Future Work



Conclusions and Future Work



Conclusions and Future Work



OXFORD e-Research CENTRE

Conclusions and Future Work

- GPU wins hands-down. **At the moment (and for the foreseeable future)!**
- ***Shared Memory Algorithm achieves between 60-70% of peak performance.***

- Next project will be to test Intels MIC (Many Integrated Core) chip – 32 in-order cores, 4 threads per core 512 bit SIMD units running a 1024 bit ring bus.

- OpenCL Algorithm, Dan Curran / Simon McIntosh-Smith (Bristol): Initial results are currently 2x slower than NVIDIA CUDA Code.

- New ARTEMIS (sub £5K) hardware with the aim of processing a full 3.2GB/s single station data stream in procurement.

Is a GPU based back-end to the Blue Gene/P at the Central Processing (CEP) facility feasible??

Acknowledgments and Collaborators

GPU de-dispersion : http://www.oerc.ox.ac.uk/research/wes

ARTEMIS : http://www.oerc.ox.ac.uk/research/artemis

University of Malta

Alessio Magro – MDSM

University of Oxford

Mike Giles (Maths) – Cuda, GPU algorithms.

Aris Karastergiou (Physics) – ARTEMIS, Astrophysics, Experimental Work.

Kimon Zagkouris (Physics) – Astrophysics, Experimental Work.

Chris Williams (OeRC) – RFI Clipper, Data pipeline.

Ben Mort (OeRC) – Data Pipeline, pelican.

Fred Dulwich (OeRC) – Data Pipeline, pelican.

Stef Salvini (OeRC) – Data Pipeline, pelican.

Steve Roberts (Engineering) – Signal processing/detection algorithms.

University of Bristol

Dan Curran (Electrical Engineering) – OpenCL work.

Simon McIntosh Smith (Electrical Engineering) – OpenCL work.

OXFORD e-Research CENTRE