



Signal detection and data processing for LOFAR using GPUs

Wes Armour
Mike Giles
Oxford e-Research Centre

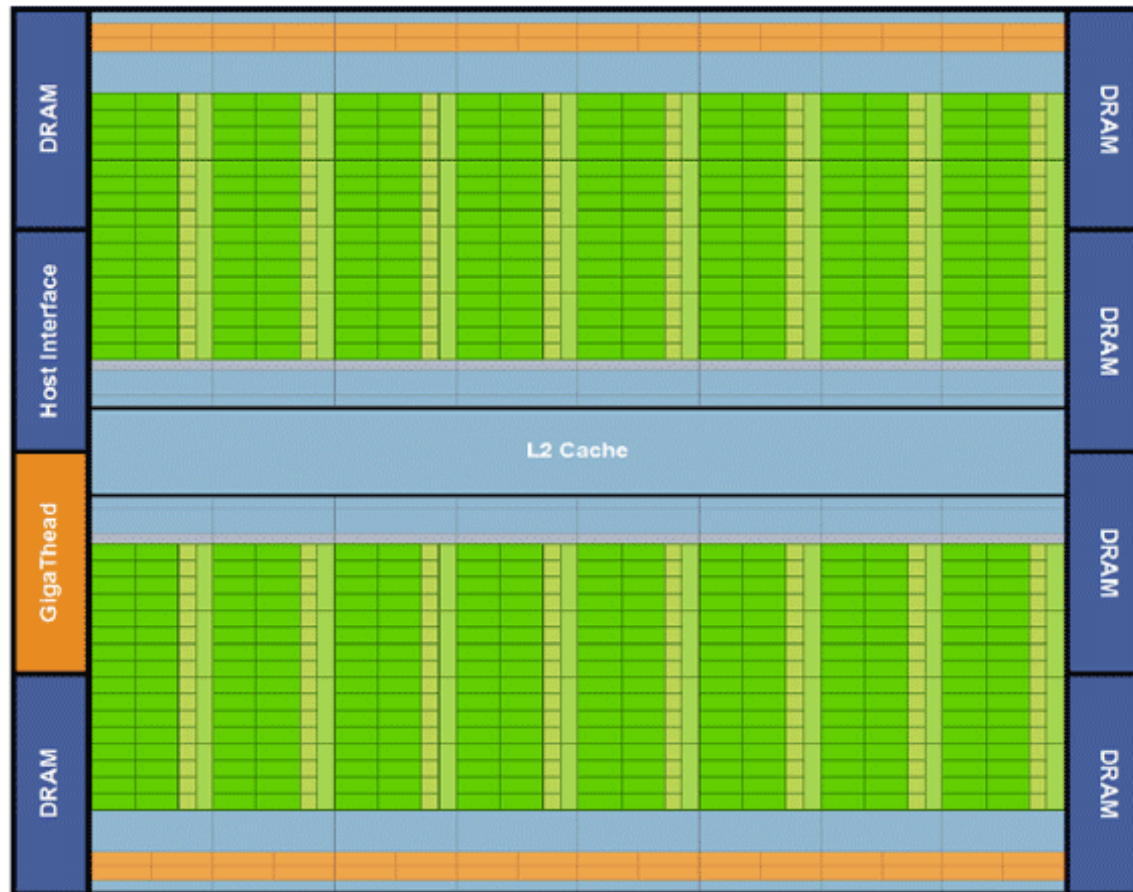
8th July 2011

Why use GPUs ?

Overview of current GPUs - Fermi

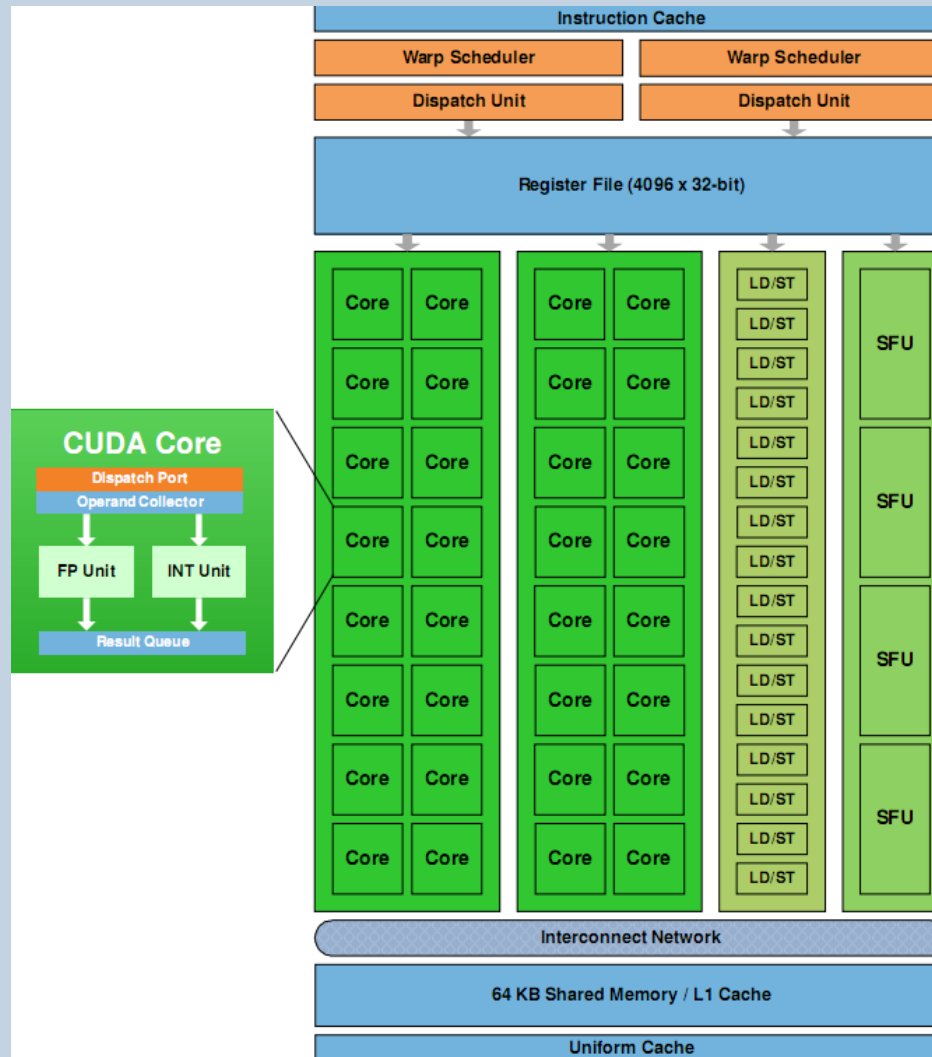
- Fermi released 2010
- 32/48 cores per Streaming Multiprocessor (SM) (sm_20/sm_21)
- Configurable 16/48K or 48/16K L1 cache / shared memory (per SM)
- New L2 cache – 768K
- GigaThread – Concurrent kernel execution (16)
- ECC – slows things down, 10% - 25%
- Simple to use extensions - CUDA for C, C++ & FORTRAN

Fermi Design



Fermi's 16 SM are positioned around a common L2 cache. Each SM is a vertical rectangular strip that contain an orange portion (scheduler and dispatch), a green portion (execution units), and light blue portions (register file and L1 cache).

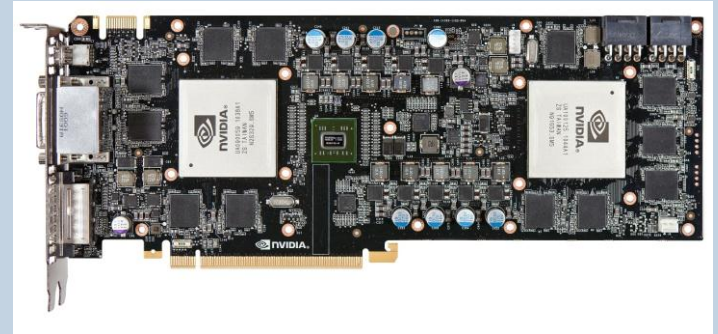
Fermi Streaming Multiprocessor (SM) design



Latest Fermi based cards

- GeForce GTX 590 – 2x GF110 GPUs
- 6.8 GFLOPS / watt
- Price point ~ £600

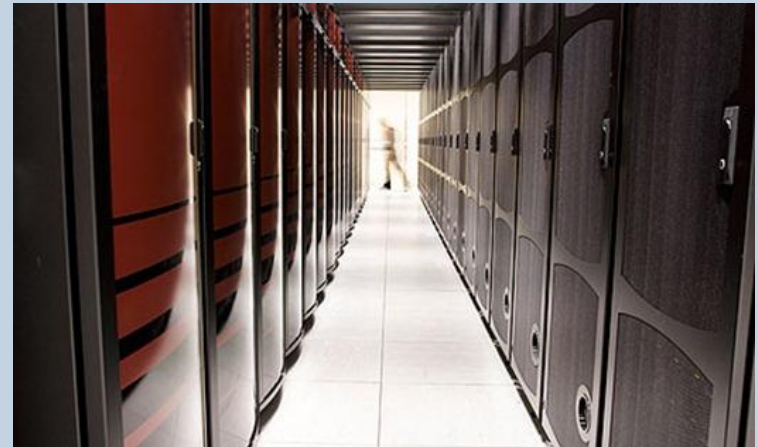
- M2090 – 1x GF110 GPU, 16 SMs,
512 cores.
- 6.8 GFLOPS / watt
- Price point ~ £2.5K



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



Radio Telescopes

LOFAR – Low Frequency Array for Radio Astronomy

- The largest radio telescope ever built - 10000+ omni-directional antennas and 77 larger stations.
- Distributed across Holland, Germany, UK, France and Sweden.
- Direction of observation is chosen by phase delays between antennas – “Aperture Synthesis”
- Can observe in several directions simultaneously - multi-user operation.
- “Path finder” for the SKA – Square Kilometre Array.

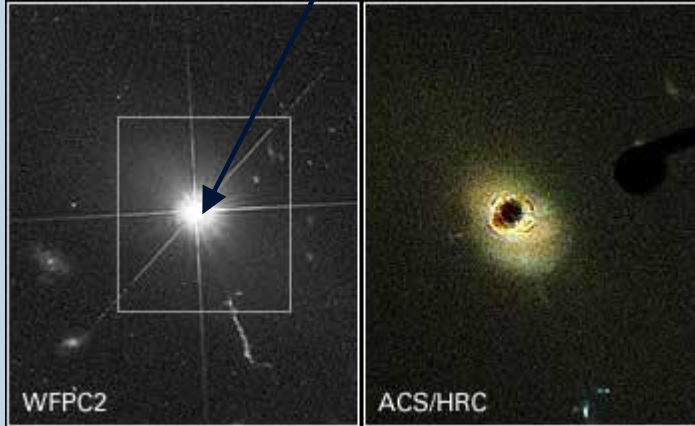


Chilbolton, UK

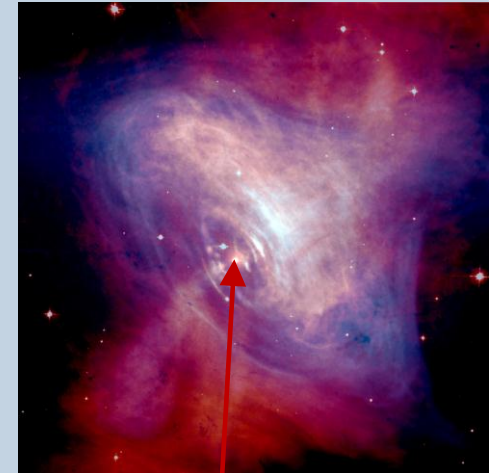
Dispersion

Radio Astronomy and Radio Transients

Quasars – Energetic region of a distant galactic core, surrounding a supermassive black hole



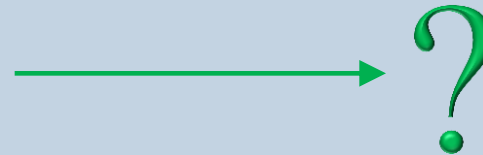
NASA and J. Bahcall (IAS)



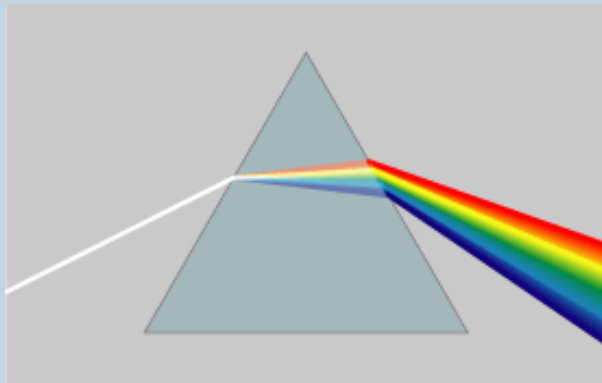
Hester et al.

Pulsars – Magnetized, rotating neutron stars. Emit synchrotron radiation from the poles, e.g. Crab Nebula

RRATS – Rotating Radio Transients. Short, bright **irregular** radio pulses. Discovered 2006

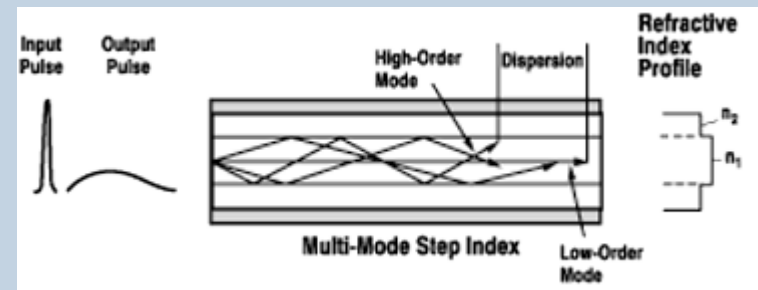


Dispersion



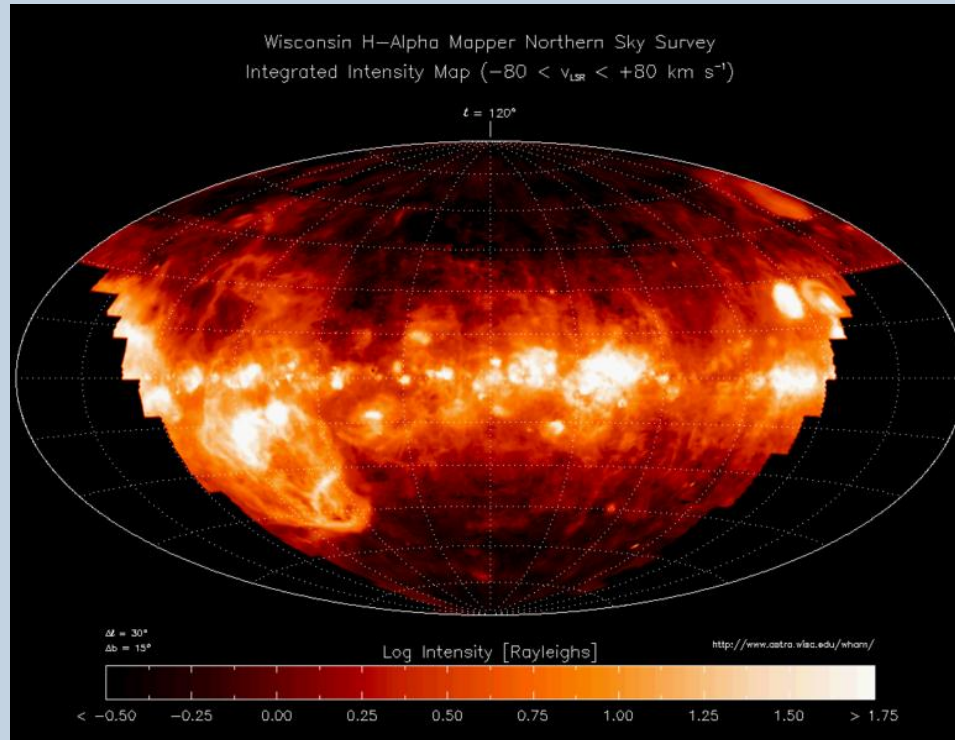
Chromatic dispersion is something we are all familiar with. A good example of this is when white light passes through a prism.

Group velocity dispersion occurs when pulse of light is spread in time due to its different frequency components travelling at different velocities. An example of this is when a pulse of light travels along an optical fibre.



Dispersion of Radio waves by the ISM

The interstellar medium (ISM) is the matter that exists between stars in a galaxy.



Haffner et al. 2003

In warm regions of the ISM ($\sim 8000\text{K}$) electrons are free and so can interact with and effect radio waves that pass through it.

The Dispersion Measure - DM

The time delay, $\Delta\tau$, between the detection of frequency f_{high} and f_{low} is given by:

$$\Delta\tau = C_{DM} \times DM \times \left(\frac{1}{f_{\text{low}}^2} - \frac{1}{f_{\text{high}}^2} \right)$$

Where C_{DM} is the dispersion constant. DM is the dispersion measure:

$$DM = \int_0^d n_e dl$$

This is the free electron column density between the radio source and observer.

We can measure $\Delta\tau$ and f and so can study DM

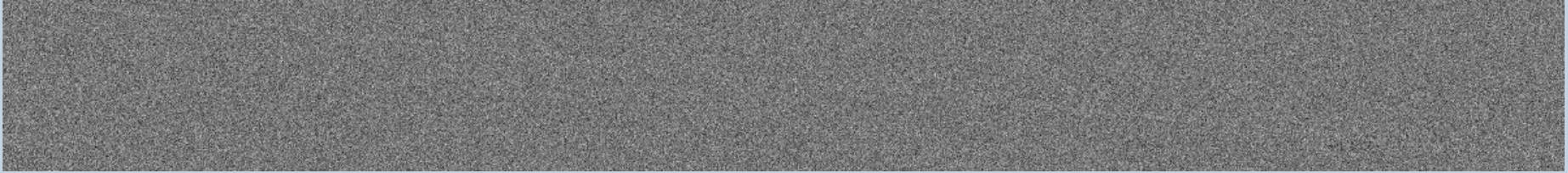
Aims of the project...

- Process a 3.2 Gb/s data stream on a single GPU.
- Allowing for a vast reduction in the cost (capital/maintenance) of compute.
- Identify appropriate areas of parallelism on both CPU and GPU.

Dedispersion

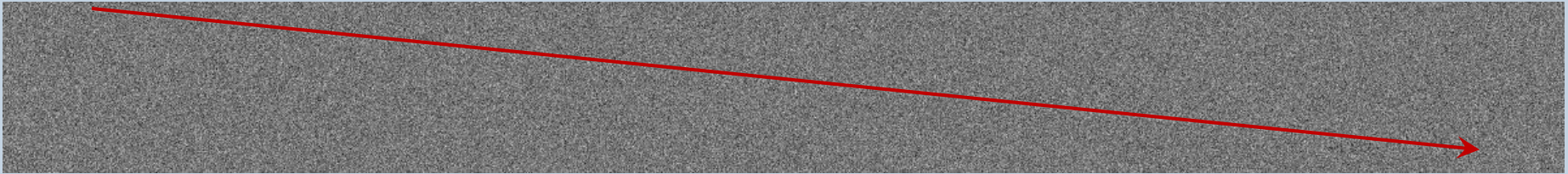
Experimental Data

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

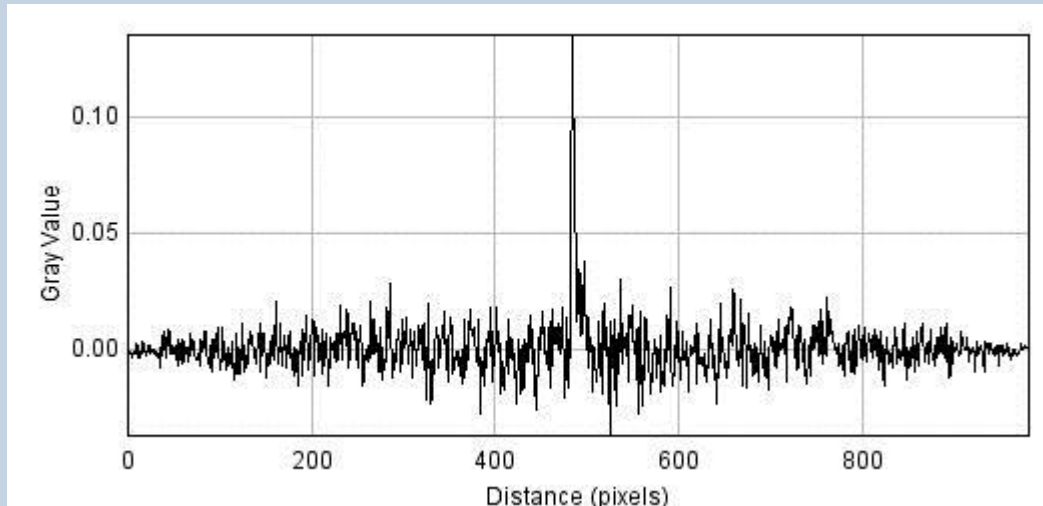


Experimental Data

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



Hence frequency channels have to be “folded”



MDSM – Alesseo, Brute force algorithm.

Algorithm 3.1 Brute-Force Dedispersion CUDA implementation

Require: nsamp, nchans, startdm, dmstep, tsamp, dm_shifts

$s \leftarrow threadIdx.x + blockDim.x * blockIdx.x$

$shift_temp = startdm + blockDim.y * dmstep / tsamp$

for samp = 0 to nsamp **do**

$samp \leftarrow samp + blockDim.x * gridDim.x$

$soffset \leftarrow s + samp$

for c = 0 to nchans **do**

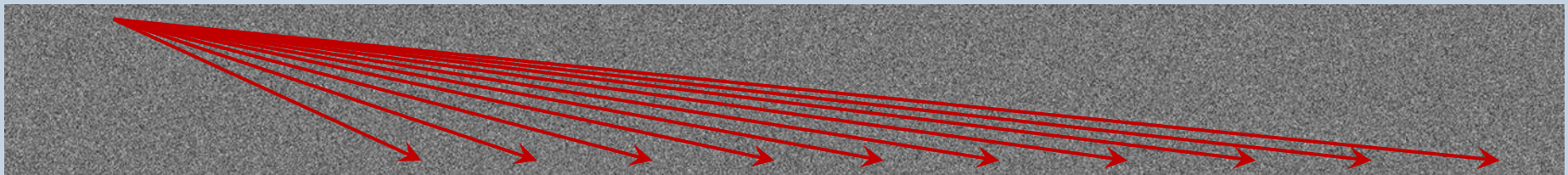
$index \leftarrow (soffset + dm_shifts[c] * shift_temp) * nchans * c$

$output[blockIdx.y * nsamp + soffset] += input[index]$

end for

end for

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



1st Stage – Memory access optimisation

- Began by profiling the code using the **Cuda Profiler**.
- Identified bottle necks in memory access patterns.
- By moving the variable that holds the running total from shared memory to a register kernel execution was **2.6x** faster.

```
__device__ float localvalue[4096];  
__global__ void opt_dedisperse_loop(...) {
```



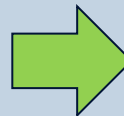
```
7: Performed Brute-Force Dedispersion 1: 1743.604004  
7: Performed Brute-Force Dedispersion 0: 1747.496582
```



```
__global__ void opt_dedisperse_loop(...) {  
    float localvalue;
```



```
7: Performed Brute-Force Dedispersion 0: 675.611206  
7: Performed Brute-Force Dedispersion 1: 677.087769
```



2.6x

Produces a **5x** speed increase in the less accurate “sub-band” method.

2nd Stage – Occupancy Optimisation

- Optimize GPU block size (number of threads) for each kernel to get the maximum occupancy of threads on the GPU.
- Investigated the optimal block size for GPU hardware versions sm_13 (Tesla) and sm_20 (Fermi).
- Found a block size of 192 on sm_13 and 256 on sm_20 produced best results.
- This produced a 1.25x speed increase.

Produces a 1.3x speed increase in the less accurate “sub-band” method.

On-going and Future Work

- Mike has suggested a new GPU friendly algorithm that we are currently implementing.
- Man made radio frequency interference can lead to false positives. So we are looking at ways of monitoring the rate of change along the DM line integral.
- More robust signal estimation using a weighted mean. Allows for systematics of the instrument to be taken into account.

Acknowledgments and Collaborators

University of Malta

Alessio Magro – MDSM

University of Oxford

- Mike Giles (Maths) – Cuda, GPU algorithms.
- Aris Karastergiou (Physics) – MDSM, Astrophysics, Experimental Work.
- Kimon Zagkouris (Physics) – Astrophysics, Experimental Work.
- Ben Mort (OeRC) – Data Pipeline, pelican.
- Fred Dulwich (OeRC) – Data Pipeline, pelican.
- Stef Salvini (OeRC) – Data Pipeline, pelican.
- Chris Williams (OeRC) – RFI Clipper Data pipeline.
- Steve Roberts (Engineering) – Signal processing/detection algorithms.