ThunderKAT 2011

Kat-7 2011 Roadmap

Simon Ratcliffe

SKA South Africa
SQUARE KILOMETRE ARRAY

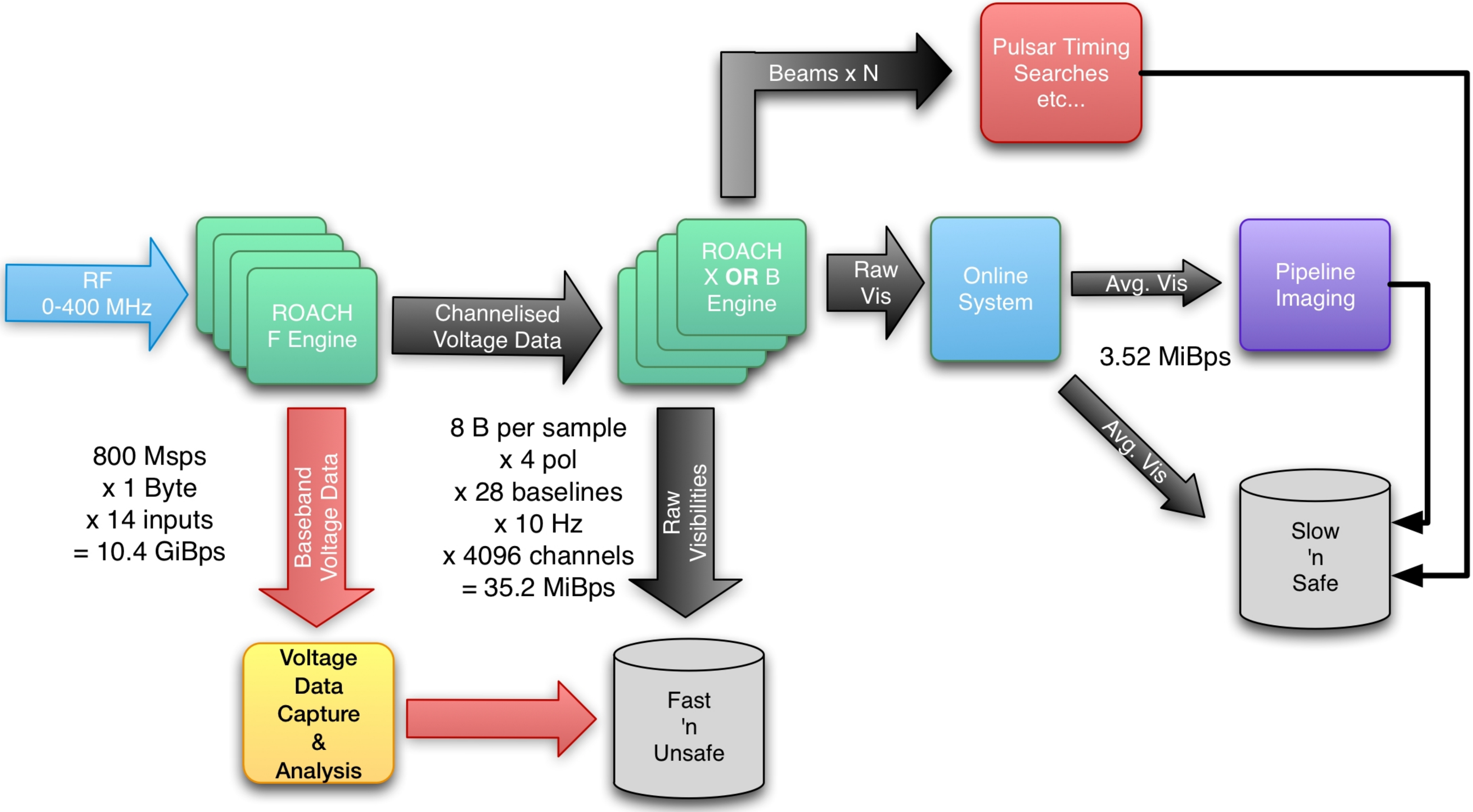
Meet the SPT – By Night

- Jasper – Keeper of 'The Voice', powerfull enough to bring the oppressive apparatus of state to it's knees.
- Ludwig – Known in street fighting circles as 'The Mathematician'
- Tom – The man with a chapeau for every occasion.
- Mattieu – Counter insurgency and black ops.
- Tshaks – Prodigious footballing talent. Secret weapon next time we engage Astron for supremacy.
- Simon – Just finished making more people, hence grumpy demeanour and incoherent (synchrotron) thought processes.

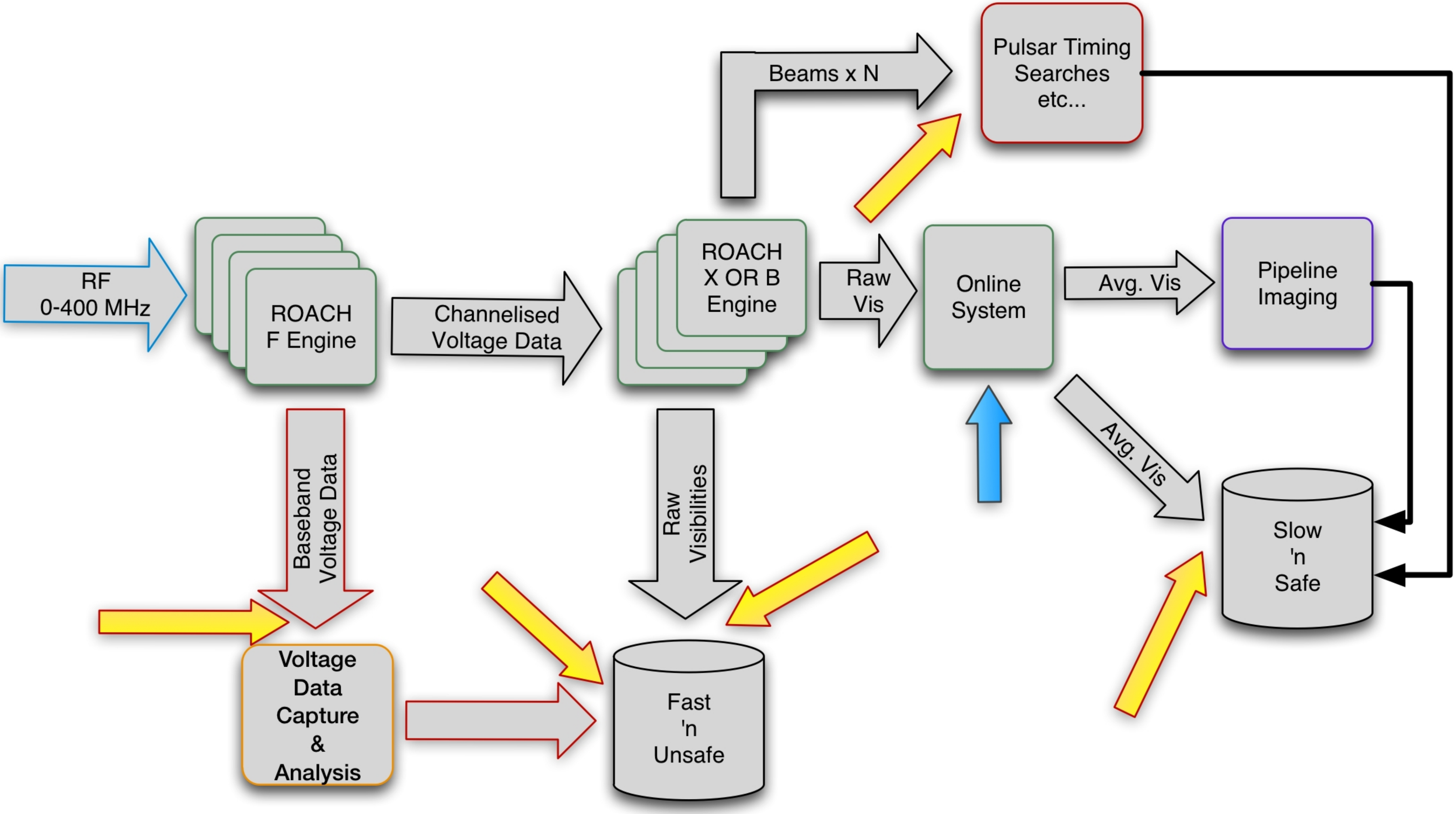
Meet the SPT – By Day

- Jasper – Chief task is herding the Kat's. In lulls between management tasks is office DiFX and MeqTree expert.
- Ludwig – Algorithm development and general mathematical guru.
- Tom – Archive, data storage and general HPC.
- Mattieu – Pol cal, layout optimisation and any other tasks that fit into his 'Master Plan'.
- Tshaks – RFI, particularly detection and removal of satellite contribution.
- Simon – Architecture, HPC, GPU optimisation and outreach activities.

Kat-7 Signal Path



Kat-7 Signal Path - Spigots (SPEAD)



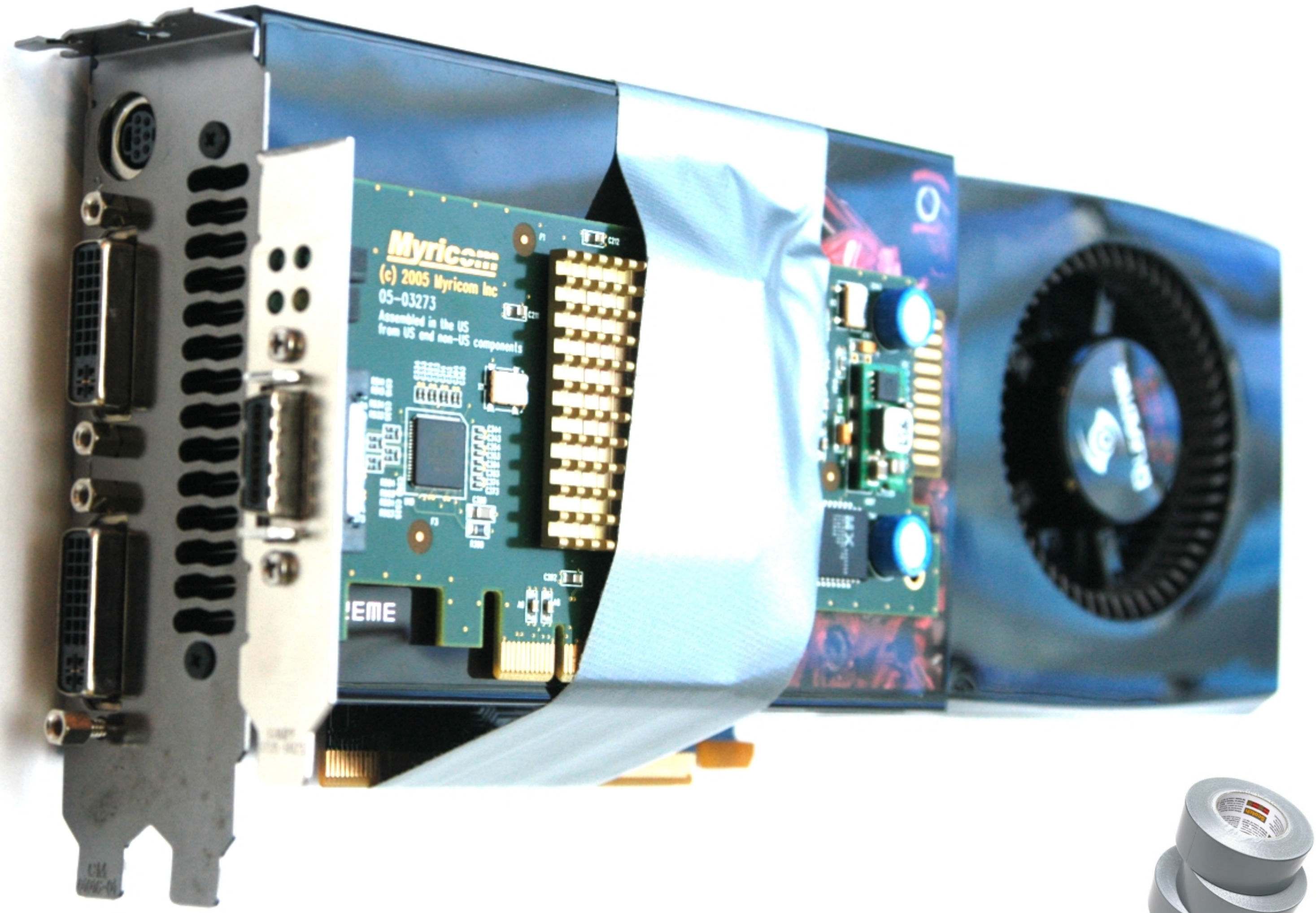
Online System

- The online system receives raw visibilities from the correlator at a sufficiently high dump rate to facilitate the following (currently 10 Hz):
  - Continuous Tsys calculation
  - RFI Flagging
  - Baseline dependent time averaging (not for K7)
- The resultant visiblities + cal data + flagging are written to disk in the safe & slow archive.
- One or more multicast data streams are produced for downstream consumers such as standard pipeline and 3rd party processors.

Flagging

- On QA Return
  - Bad antenna / polarisation
  - Timing synch
  - Etc...
- RFI Flagging
  - Critical to avoid costly rerun through data set.
  - Simple thresholding
  - Known sky pollutants (GEO, LEO, DME, etc...)
  - Recursive fading-memory polynomial filters (we have an IBM System-S implementation, porting to GPUs soon)

Our Basic Building Block



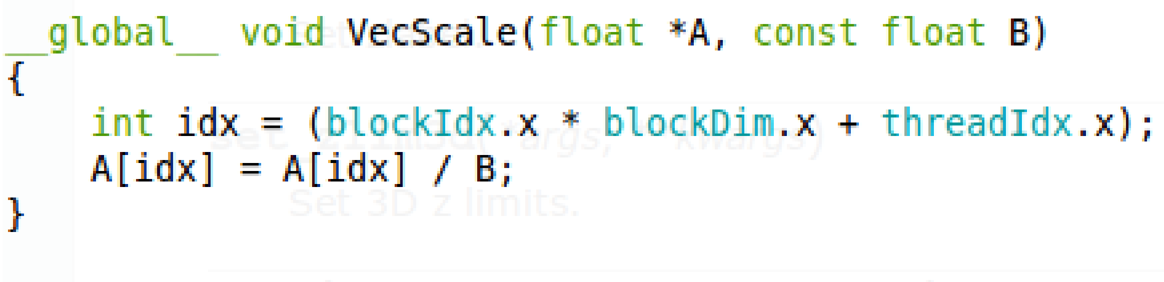
=

Why GPUs

- A single GPU is a good match for current performance networking and has the I/O to keep up with data from both 10 GbE and QDR infiniband.
- Still riding good performance curves and HPC support is getting better all the time.
- Two major players (Nvidia and ATI) can help with vendor lock in (as long as you stick with OpenCL and not CUDA).
- The ecosystem of tools, particularly debuggers is rapidly improving (see Nvidia Nsight for an example of good these are getting)
- IRQ affinity under linux is helping to support multiple GPUs and NICs per machine.
- New developments (AMD Fusion almost a reality) will bring the GPU ever closer to the CPU memory bus, improving throughput and interoperability.

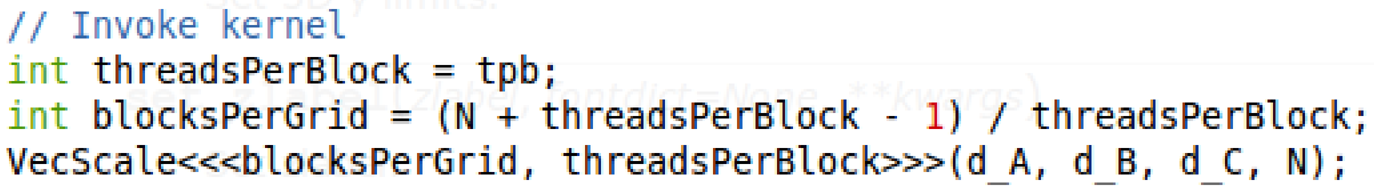
CUDA – Hello World

Kernel – Spawned and executed per GPU thread



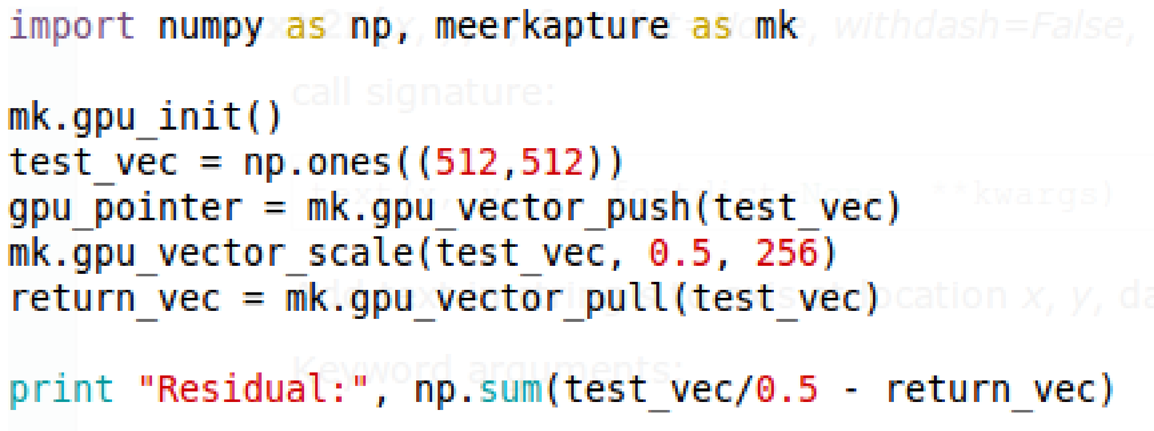
```
__global__ void VecScale(float *A, const float B)
{
    int idx = (blockIdx.x * blockDim.x + threadIdx.x);
    A[idx] = A[idx] / B;
}
```

CPU code – C with CUDA markup



```
// Invoke kernel
int threadsPerBlock = tpb;
int blocksPerGrid = (N + threadsPerBlock - 1) / threadsPerBlock;
VecScale<<<blocksPerGrid, threadsPerBlock>>>(d_A, d_B, d_C, N);
```

Python wrapper – Do things the easy way :)

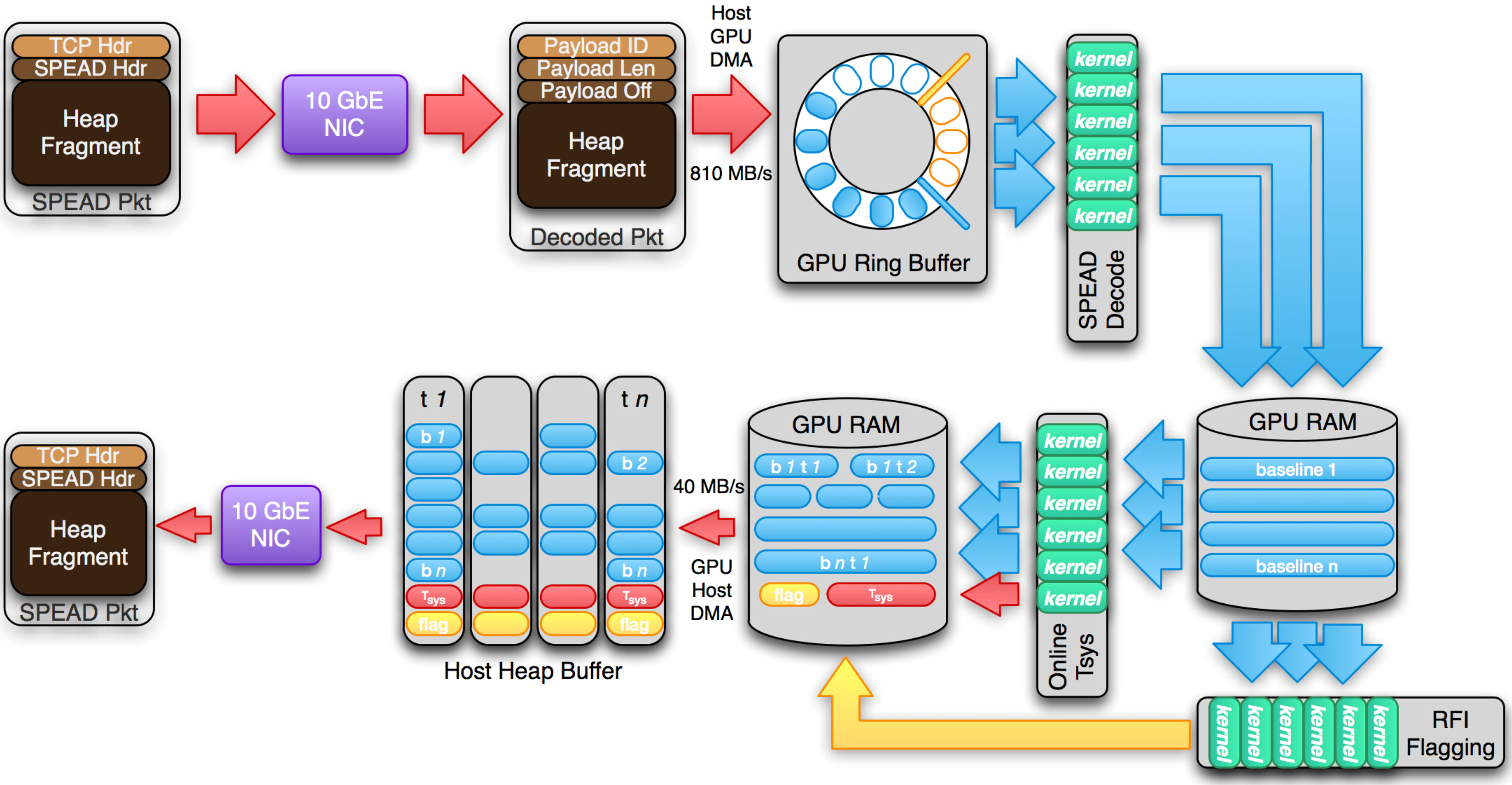


```
import numpy as np, meerkapture as mk

mk.gpu_init()
test_vec = np.ones((512,512))
gpu_pointer = mk.gpu_vector_push(test_vec)
mk.gpu_vector_scale(test_vec, 0.5, 256)
return_vec = mk.gpu_vector_pull(test_vec)

print "Residual:", np.sum(test_vec/0.5 - return_vec)
```

Online System – On the GPU



The Glue....

RUBBER INFUSED
SPEAD
SUPER GLUE-3
Max Flex Gel
FLEXIBLE
waterbestendig
résistante à l'eau
Alle materialen
Tous matériaux
LOCTITE SUPER GLUE-3
Flex Gel Max
SECONDELIJM/COLLE INSTANTANÉE 10g
Henkel

SPEAD

- Streaming Protocol for Exchanging Astronomical Data
- Joint development between SKA South Africa and UC Berkeley as part of the CASPER collaboration.
- Designed to handle a wide variety of astronomical data including voltage, visibility, and sensor data.
- Standard output data format for ROACH based correlators.
- Aim is to have a single coherent protocol throughout the entire processing chain (i.e. from digitisation to imaging)

SPEAD

- Specification is currently in revision L, update coming soon (CASPER workshop next week)
- Reference Python implementation available from:

http://github.com/sratcliffe/PySPEAD.git

- GPU accelerated en/decode not in the public release yet. Still deciding between CUDA and OpenCL.
- Promises to have a fairly large number of users which always helps !

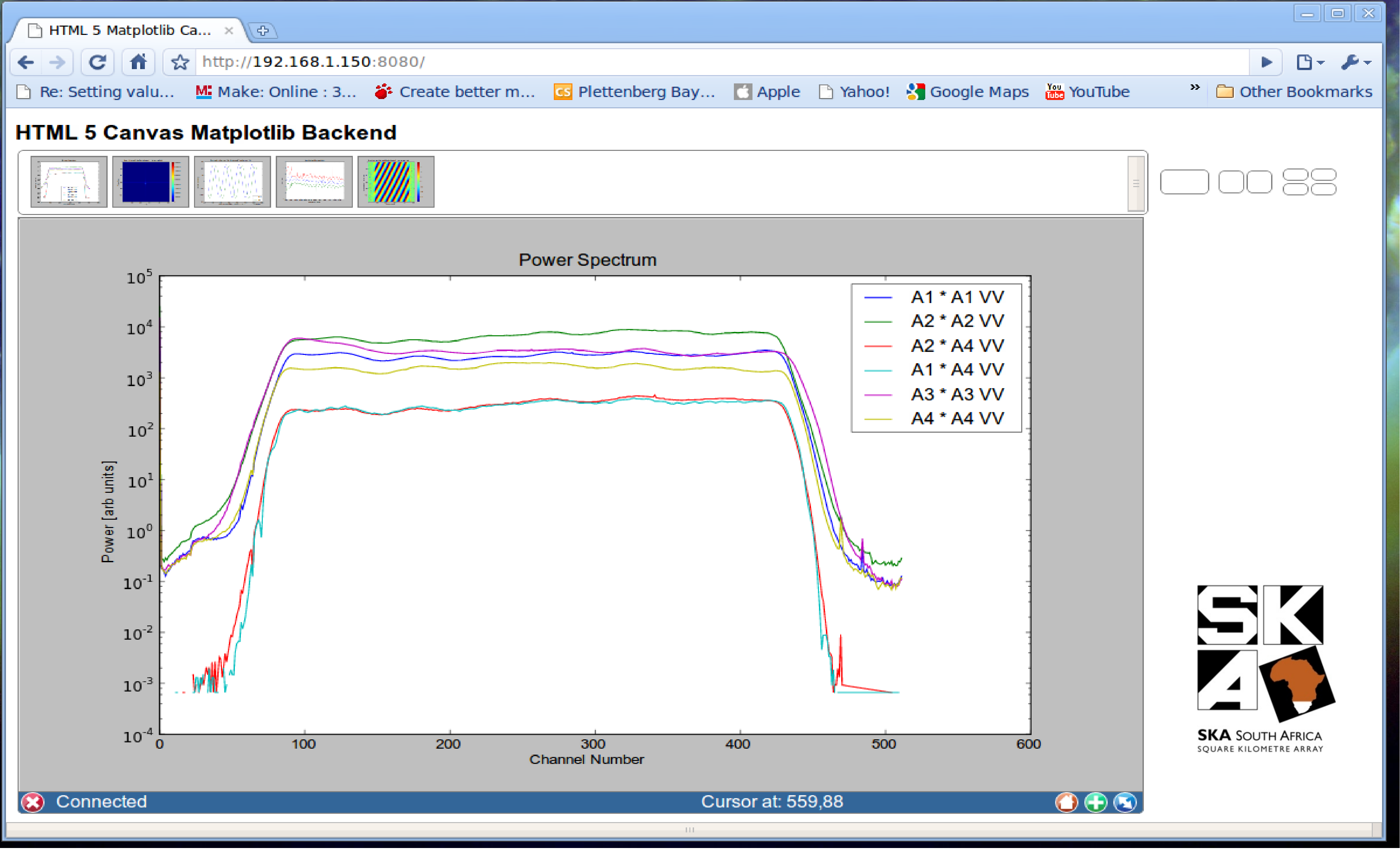
Online System – Data Formats

- Two standard output flavours: archive and stream.
- Stream output takes form of one or more SPEAD streams which encapsulate vis data, flagging, weights and other meta data.
- End user of SPEAD stream basically sees an evolving dictionary of Numpy arrays containing data for current timestamp.
- Archival format uses HDF5 as underlying file format.
- Currently a superset of both SDM and MSv2 to allow seamless export to either (currently only MS support, SDM if requested).

Online System – Signal Displays

- Produces reduced rate signal display data for QA and engineering testing.
- Range of matplotlib based displays from power user to read only canned plots.
- Uses internally developed HTML5 backend for Matplotlib to delivery displays remotely.
- PI will have access to select displays from outside the Cape Town office.

KAT-7 Signal Displays



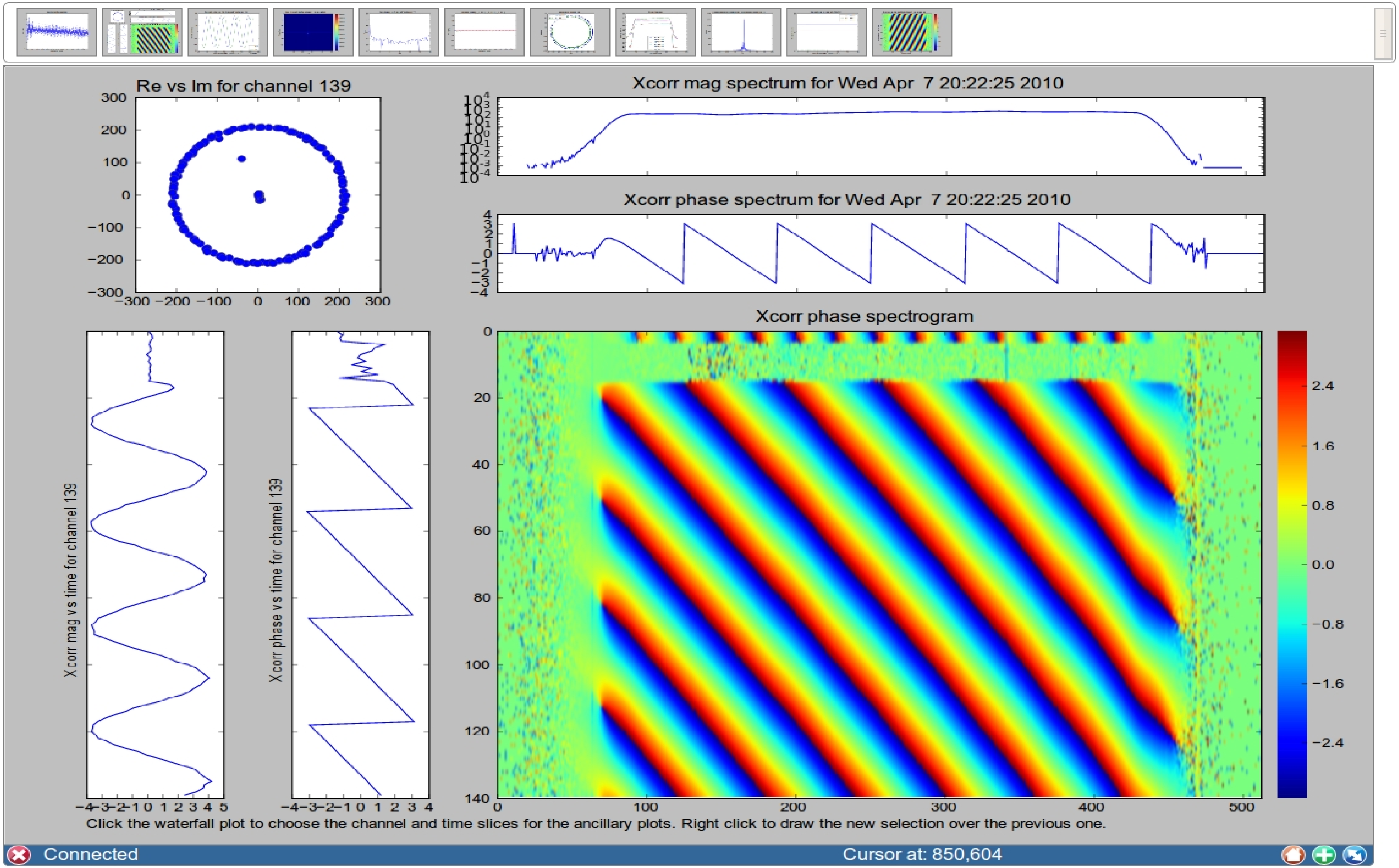
HTML 5 Canvas Matplotlib Backend

Connected

Cursor at: 559,88

KAT-7 Signal Displays

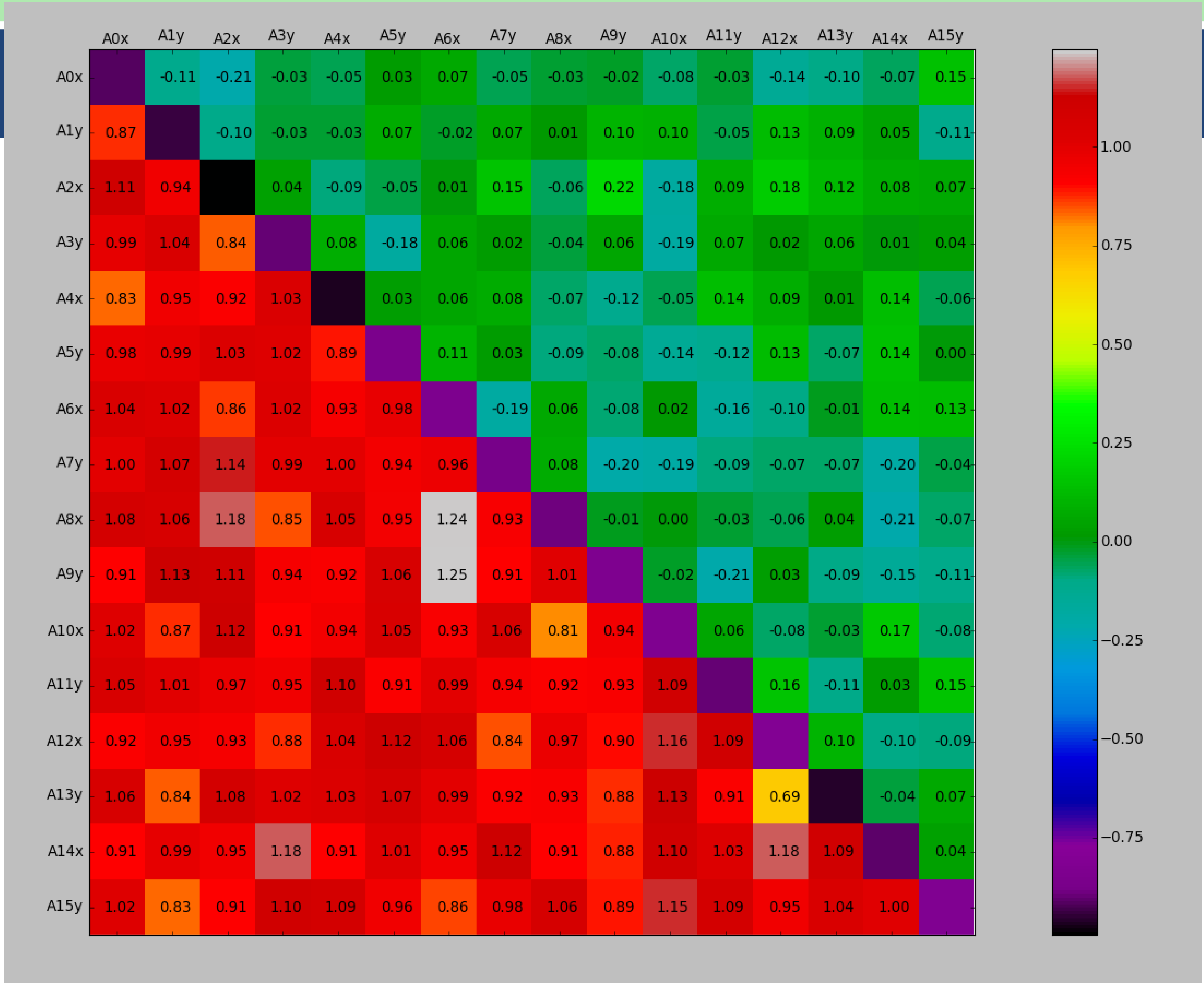
HTML 5 Canvas Matplotlib Backend



Click the waterfall plot to choose the channel and time slices for the ancillary plots. Right click to draw the new selection over the previous one.

Connected

Cursor at: 850,604



	A0x	A1y	A2x	A3y	A4x	A5y	A6x	A7y	A8x	A9y	A10x	A11y	A12x	A13y	A14x	A15y
A0x		-0.11	-0.21	-0.03	-0.05	0.03	0.07	-0.05	-0.03	-0.02	-0.08	-0.03	-0.14	-0.10	-0.07	0.15
A1y	0.87		-0.10	-0.03	-0.03	0.07	-0.02	0.07	0.01	0.10	0.10	-0.05	0.13	0.09	0.05	-0.11
A2x	1.11	0.94		0.04	-0.09	-0.05	0.01	0.15	-0.06	0.22	-0.18	0.09	0.18	0.12	0.08	0.07
A3y	0.99	1.04	0.84		0.08	-0.18	0.06	0.02	-0.04	0.06	-0.19	0.07	0.02	0.06	0.01	0.04
A4x	0.83	0.95	0.92	1.03		0.03	0.06	0.08	-0.07	-0.12	-0.05	0.14	0.09	0.01	0.14	-0.06
A5y	0.98	0.99	1.03	1.02	0.89		0.11	0.03	-0.09	-0.08	-0.14	-0.12	0.13	-0.07	0.14	0.00
A6x	1.04	1.02	0.86	1.02	0.93	0.98		-0.19	0.06	-0.08	0.02	-0.16	-0.10	-0.01	0.14	0.13
A7y	1.00	1.07	1.14	0.99	1.00	0.94	0.96		0.08	-0.20	-0.19	-0.09	-0.07	-0.07	-0.20	-0.04
A8x	1.08	1.06	1.18	0.85	1.05	0.95	1.24	0.93		-0.01	0.00	-0.03	-0.06	0.04	-0.21	-0.07
A9y	0.91	1.13	1.11	0.94	0.92	1.06	1.25	0.91	1.01		-0.02	-0.21	0.03	-0.09	-0.15	-0.11
A10x	1.02	0.87	1.12	0.91	0.94	1.05	0.93	1.06	0.81	0.94		0.06	-0.08	-0.03	0.17	-0.08
A11y	1.05	1.01	0.97	0.95	1.10	0.91	0.99	0.94	0.92	0.93	1.09		0.16	-0.11	0.03	0.15
A12x	0.92	0.95	0.93	0.88	1.04	1.12	1.06	0.84	0.97	0.90	1.16	1.09		0.10	-0.10	-0.09
A13y	1.06	0.84	1.08	1.02	1.03	1.07	0.99	0.92	0.93	0.88	1.13	0.91	0.69		-0.04	0.07
A14x	0.91	0.99	0.95	1.18	0.91	1.01	0.95	1.12	0.91	0.88	1.10	1.03	1.18	1.09		0.04
A15y	1.02	0.83	0.91	1.10	1.09	0.96	0.86	0.98	1.06	0.89	1.15	1.09	0.95	1.04	1.00	

Online System – Simulations 0.1

- Point made earlier about integration of components.
- MeerKAT system engineering approach works to mitigate this very problem.
- With the CAM team we produce simulators for every system component once their thin specification has been produced.
- We refine the simulator with the developers of the component and encourage them to benchmark against it.
- At any time we can launch and test a system with a mix of real and simulated components.

Online System – ThunderKAT

- For Kat-7 you can choose an integration period for the vis data. Probably practical to support 100ms to 1000s.
- Supporting up to 10 simultaneous integration periods for a variety of users should be fine.
- Simplest would be receiving data directly into Numpy array form in Python. (Perhaps imaging using AIPY ?)
- This would require processing time < integration period.
- Otherwise single integration MS is quite feasible.
- We can immediately provide Python based SPEAD simulation code.
- An entire Kat-7 simulated environment can easily be provided.

Voltage Data

- Currently we have a single pol 400 MHz baseband recorder.
- Implementation uses a GPU to perform either PFB or DDC.
- Not power or space efficient.
  - 1000W PSU
  - 10 U

Voltage Data

- RFI (the good kind) out for expanded voltage capture for Kat-7.
- System will record synchronous dual pol baseband data for all 7 antennas.
- Spec is 2 min at full band, with goal of 10.
- 10-20s memory ring buffer will allow some lookback for high time resolution follow up.
- Plenty of GPU cycles spare so triggers could be locally generated (Fly's Eye).

Voltage Data

- Data stored in raw 8-bit form with export to 2-bit VDIF format available.
- Duty cycle between events is of order 10 times the capture duration.
- Software correlation using DiFX will be available.
- For us voltage data has been a whole new ballgame. Simple things such as visualisation have had to be substantially rethought.
- Working on some Python tools for exploring large raw data sets.

Kat-7 Archive

Calibrated visibilities that pass QA will be stored indefinitely !

Holds true for MeerKAT given the shift in timelines.

Kat-7 Archive - Breakdown

- Slow and safe:
    - ~ 100 TiB
    - 1 Gbps
    - Vis data, meta data, output products
- Fast and safe:
    - ~ 30 TiB
    - 20+ Gbps
- Fast and unsafe:
    - ~ 30 TiB
    - 20+ Gbps

Kat-7 Archive – 1 year at a time

- Visibility Data:
  - 300 days nominal
  - 162.2 TiB
  - 2:1 compression straightforward
- Meta Data:
  - ~ 8 TiB
- Output Products:
  - Anything involving significant human or cpu cost.
  - ~ 12 TiB

Kat-7 Archive - Checkout

- Web interface used for searching.
- Small datasets can be downloaded directly in desired format.
- Larger datasets can be staged to specified host machine(s) from the interface.
- Datasets can be split by frequency across nodes so as to do away with a network file system.
- SPEAD stream can be instantiated (includes required metadata)

Kat-7 Archive

- Mirror of safe & slow archive will be available in Cape Town.
- Hope is to host archive at CHPC, which will then couple significant processing capacity to the archive for reprocessing and mining.
- Latency probably of order hours.

Kat-7 Archive - ThunderKAT

- Dedicated space in S&S archive for science data.
- High speed scratch space available via iSCSI mount if required.
- Science data will be replicated to Cape Town archive.
- If space can be found in Europe then a second mirror can be setup.
- Possibility of producing longer cadence images in archive instead of realtime ?

Open Questions

- What kind of calibration is needed. Is basic Tsys weight enough ?
- How would you like the data for Kat-7 and will this change with MeerKAT ? (Streams / Files)
- Does a push to higher rate imaging (say 10 or 100 Hz) buy much ?
- How does RFI flagging feed in ?
- Model for collaboration (perhaps schedule a dreaded 'Busy Week')