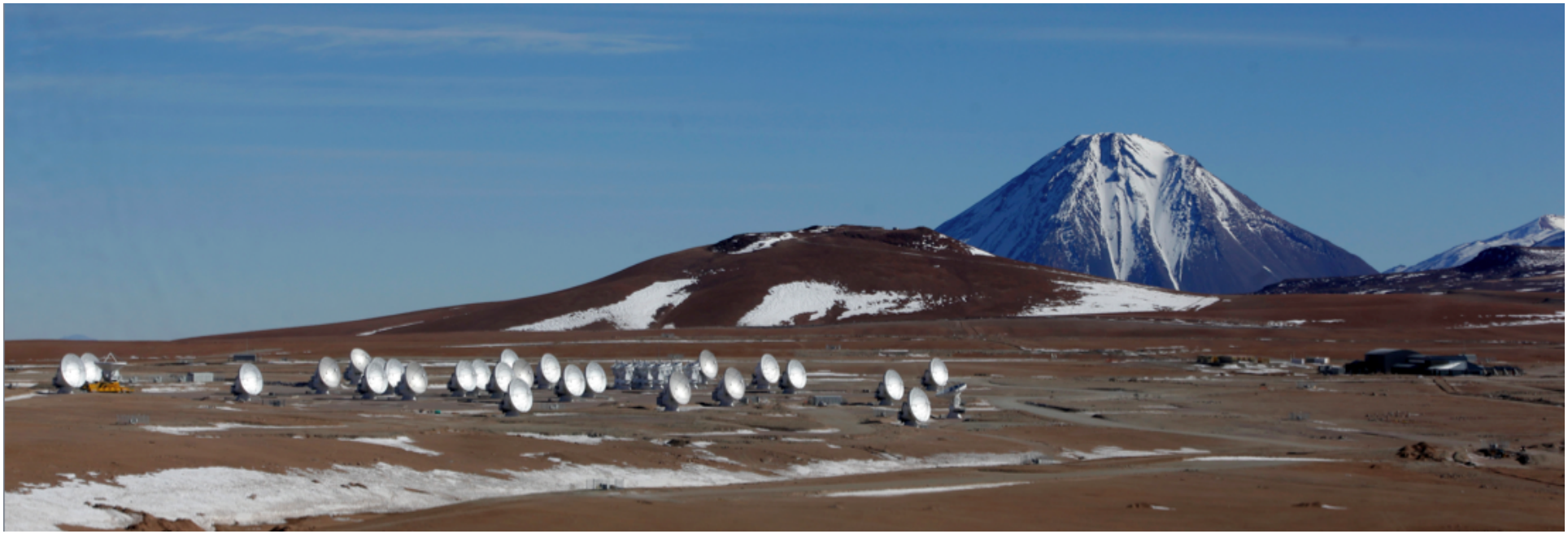
CASA 4.0, HPC, and Beyond



North American ALMA Science Center

Crystal Brogan

Atacama Large Millimeter/submillimeter Array
Expanded Very Large Array
Robert C. Byrd Green Bank Telescope
Very Large Baseline Array

What's New in CASA 4.0

(@ Nov 15)

CASA 4.0 from ALMA perspective -I

Infrastructure

- Virtual model data column (reduces calibrated datasize by 1/3)
- Conversion to SWIG binding of C++ and python (improve maintainability)

Calibration

- "nearest" option in applycal – selects calibration from nearest field on the sky with calibration available
- Linear polarization (still needs some commissioning on ALMA)
- New flagging options in applycal
- Improved logging of failed solutions
- Fluxscale is now just the relative scaling (incremental) with amplitude table
- Setjy has new and improved Solar System models

Imaging

- Threaded gridder (up to 4 cores)
- Per plane restoring beams (needed for wide bandwidths)
- Parallel clean (cubes and simple continuum)
- Additional scaling parameters in Feather task

CASA 4.0 from ALMA perspective -II

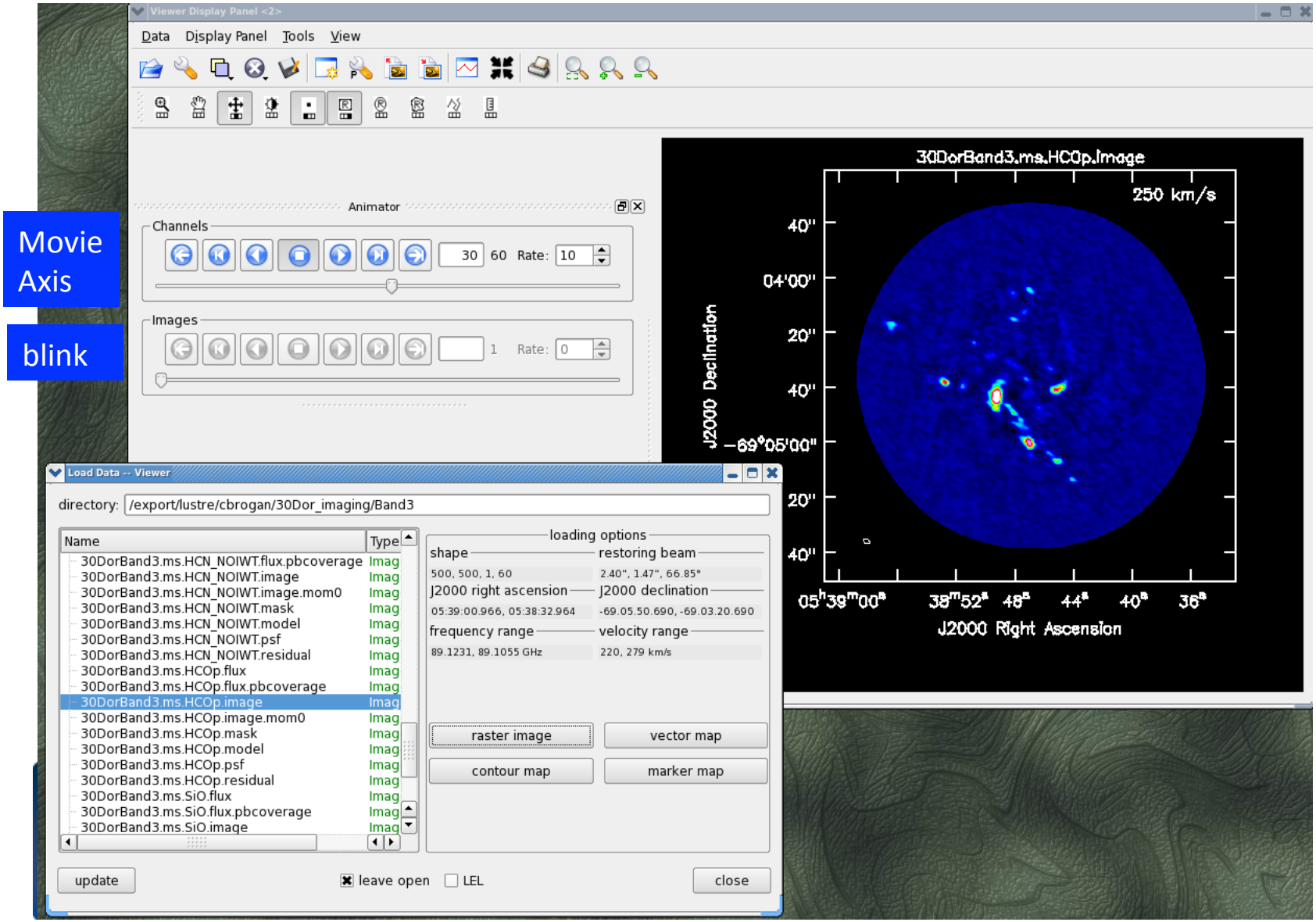
Image Analysis
- Support for per plane beams
- New imsubimage task (simplier interface compared to immath usage)
- Makemask task

PlotMS
- ASCII Dump of selected points
- Scriptable/ Auto sizing of plotted points

Viewer (Rasters)
- OTF spectral velocity gridding
- FITS and CASA images show data summary when selected
- Additional marker types for regions
- Clean up of new region format bugs

Single Dish (EA CASA staff)
- Improved calibration mode for OTF data
- Restfreq parameter available in most tasks
- Sdbaseline can now use Chebyshev polynomials
- Channel flagging specifying frequency or velocity (not just channel)

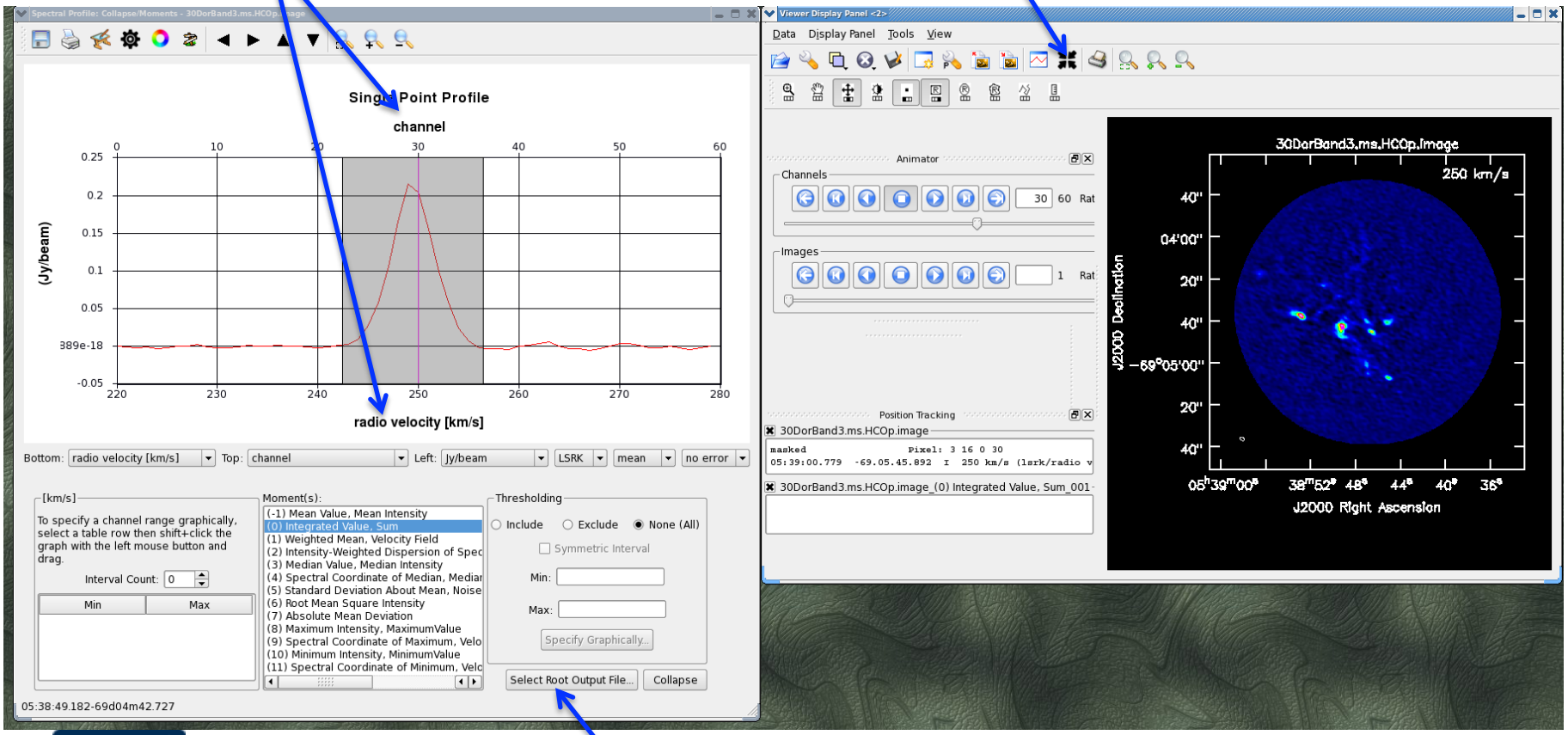
Image Summary and Decoupling of Movie axis and "Blink"



Create Moment Maps on the Fly and show two spectral axes at once

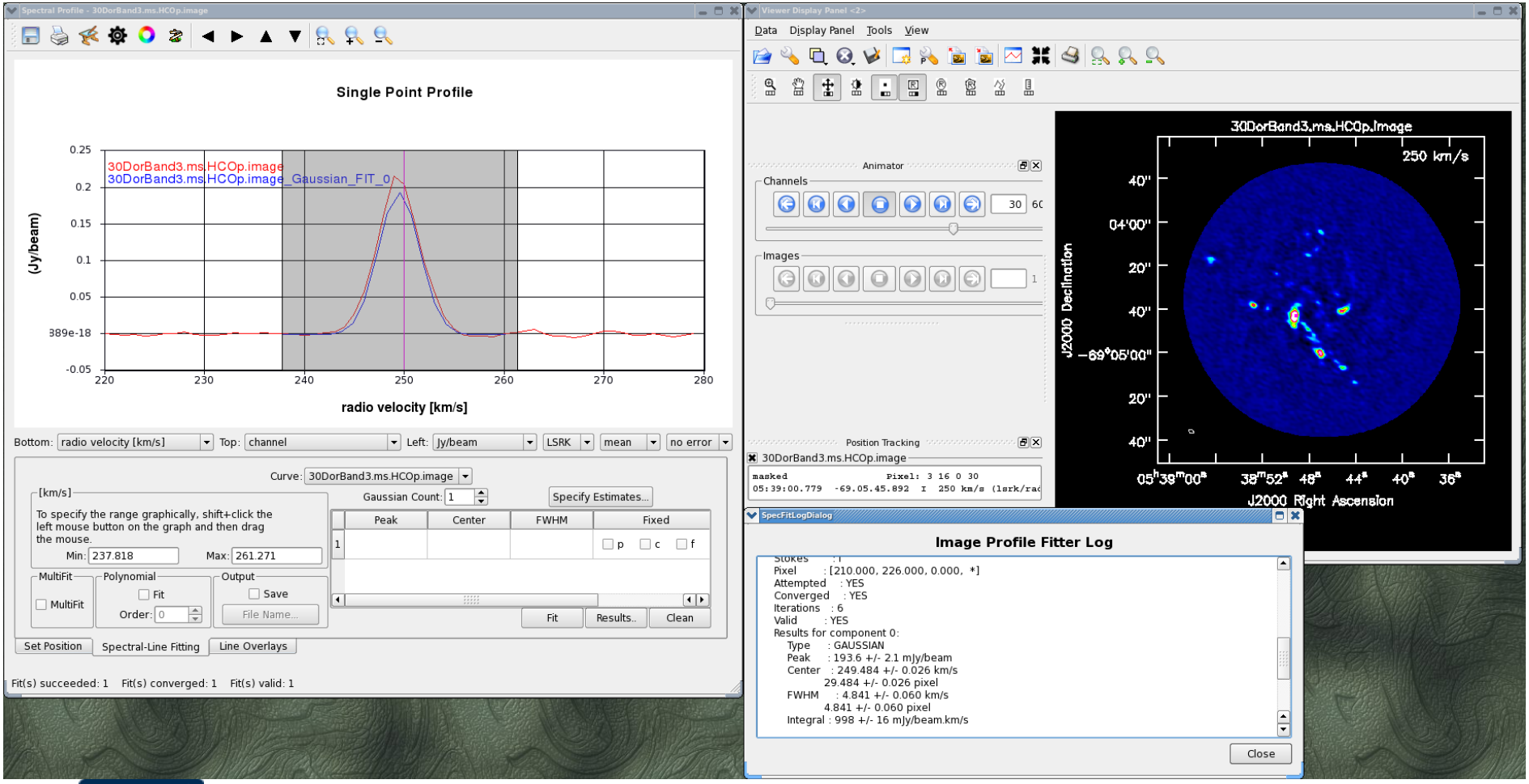
Two spectral axes displayed at once

Open moment map tool

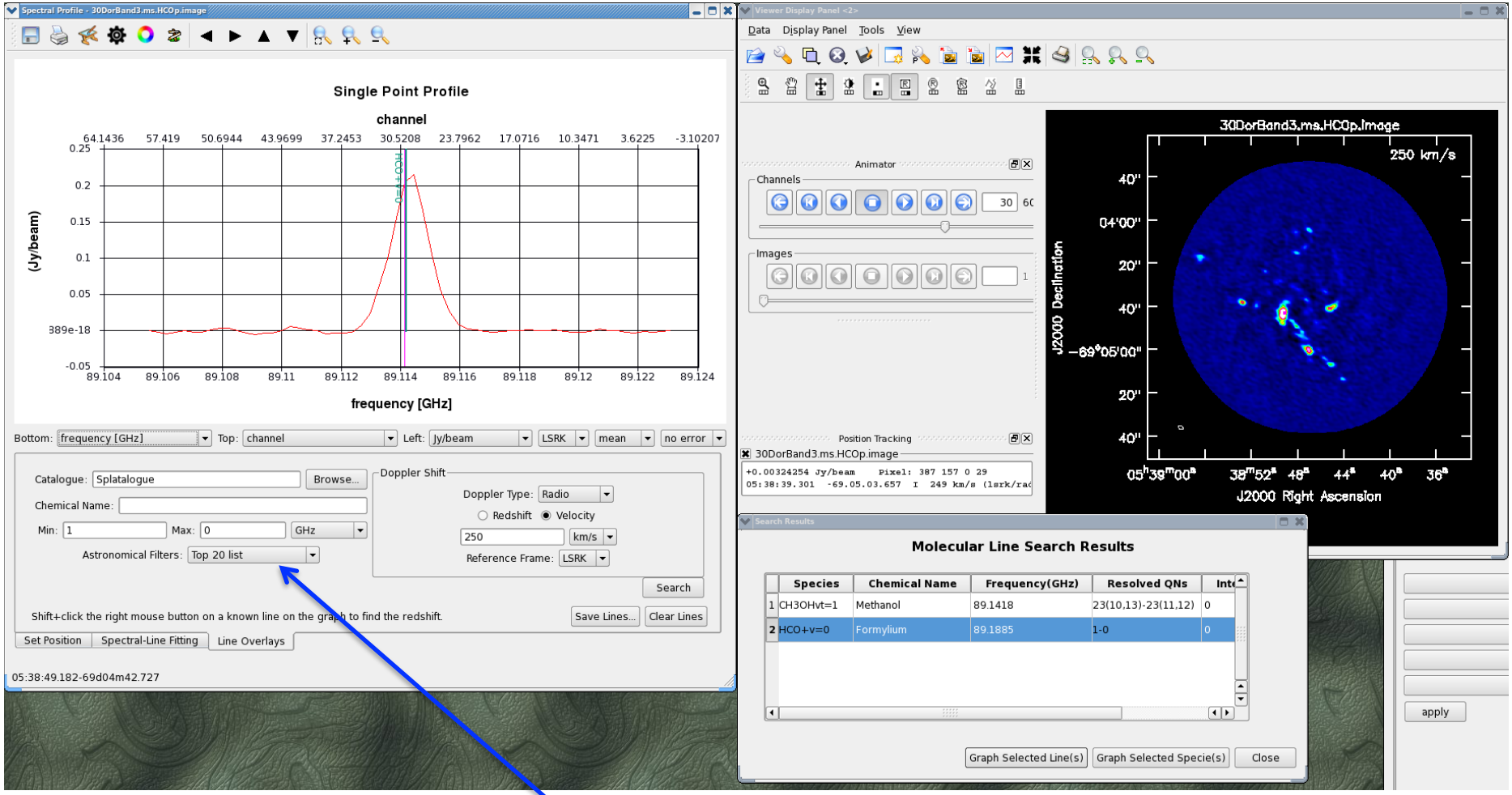


Save output or just view it

Fit Gaussians and Save Output



Interactively Id Lines (first step)



Same filters available in the OT

What to Expect in CASA 4.1

(@ June 2013)

Planned Visualization Improvements

- Increased support for publication quality plots.
  Annotations, axis controls, color bars, labeling, etc.

- Improved/ more interactive line identification querying Splatalogue with astronomical filters.

- OTF regridding among cubes with different astrometric and velocity gridding.

- More sophisticated interactive spectral line fitting and access to batch processing

- Support for P-V diagrams

- Improved ability to call viewer from the command line.
  Expand existing capabilities to minimize clicking to perform common tasks.

Re-factor of Clean

Required to:

- Improve maintainability! This is currently a serious issue
- Make the look and feel the same when using different modes (i.e. logging info is currently very different between modes)
- Improve management of minor and major cycles
- Make it easier to implement combined modes like MFS with nterms>1 AND mosaicing (a much requested capability)
- Easier binding of all current functionality to parallel clean

Performance
(current status)

Why we're worried = large data volumes

- ALMA is currently designed to allow archiving of up to 60 MB/s of raw visibility data (16-Bit), i.e. an ASDM.
- System was designed assuming a 6 MB/s average

	Antennas	Int Time	MB/s	One Hour ASDM
4 TDM	16	1 s	0.5	1.8 GB
4 FDM	16	6 s	2.3	8.3 GB
4 TDM	50	1 s	4.8	17.3 GB
4 FDM	50	6 s	23.9	86.0 GB

The MS is 32-Bit, so these numbers are doubled for the processing size

- 6s averaging for FDM is self-imposed to keep size down but allow for high dynamic range self-calibration
- A typical project can have several executions that must be combined for imaging

Hardware & Terminology

CASA has focused on two "standard" systems:

Workstation	Cluster
• Multi-core system	• Many multi-core nodes
• Local disk	• High performance network file system (Lustre)
• Single shared memory	• No shared memory access

Terminology:

- Core: A single processing element that reads and executes instructions.
- Node: A single host, a modern node usually has multiple cores.
- Engine: A single instance of CASA connected through a messaging interface, in most cases a single engine consumes a single core.
- Multi-Process: Many independent processes each with their own processing space.
- Multi-Threaded: A single process with multiple threads of execution, multiple threads can share the same memory object.

Type of Disk Space Matters (a lot!)

- Many processing steps require traversing the entire data set and are thus data Input/Output limited.

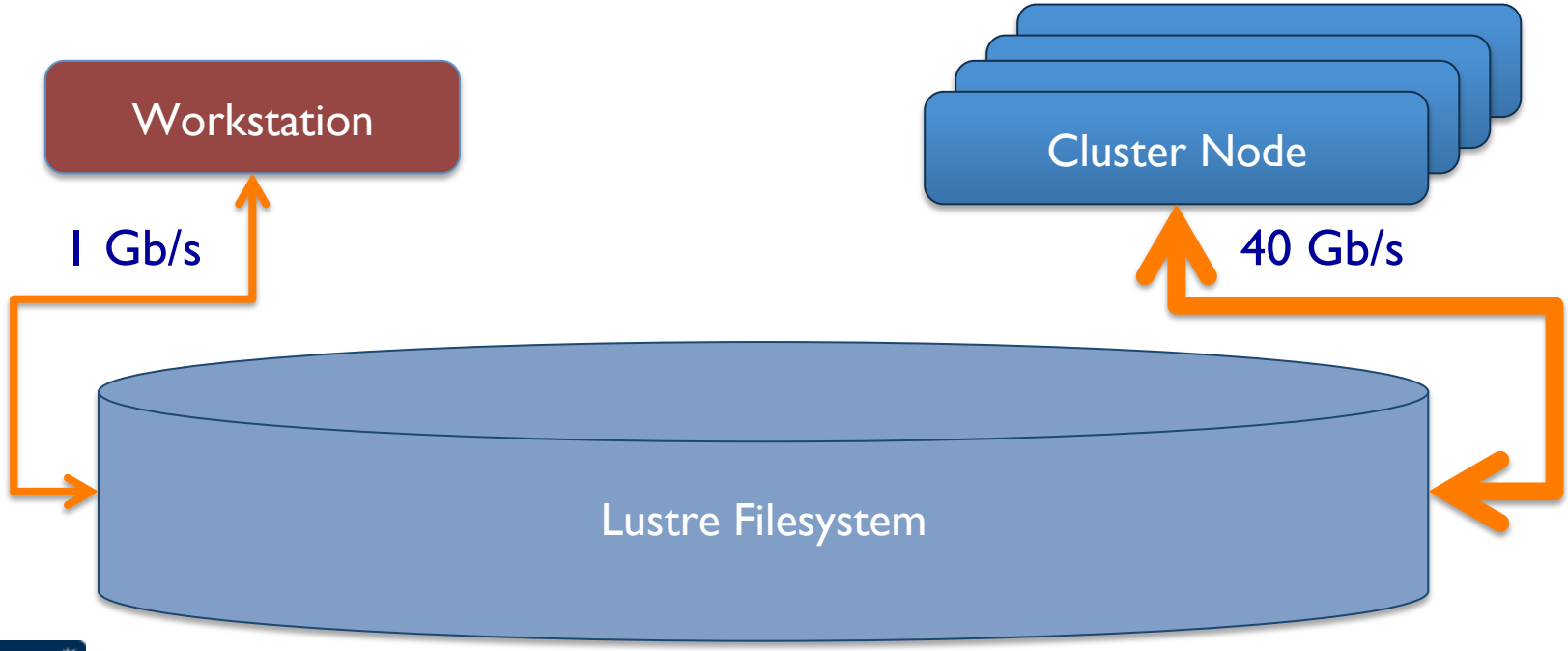
	Peak Data Rate	OS Realized Data Rate	Time for 500 GB file
SATA Disk	115 MB/s	60 MB/s	2h 22 m
Raid	200-500 MB/s	375 MB/s	22 m
Lustre (10 GB)	1.2 GB/s	700 MB/s	12 m

- Even imaging [in the simplest case] is IO limited, requiring about 50 MB/s per core to prevent the CPU from incurring wait states.

https://science.nrao.edu/facilities/evla/data-processing/hardware-recommendations

Lustre

- Running against Lustre can provide significant performance improvement, but how its attached matters.
- How the data is arranged on Lustre impacts performance as well.



Parallel Approach in CASA

An embarrassingly parallel workload is one for which little or no effort is required to separate the problem into a number of parallel tasks, this is often the case where there is exists no dependency between the parallel tasks.

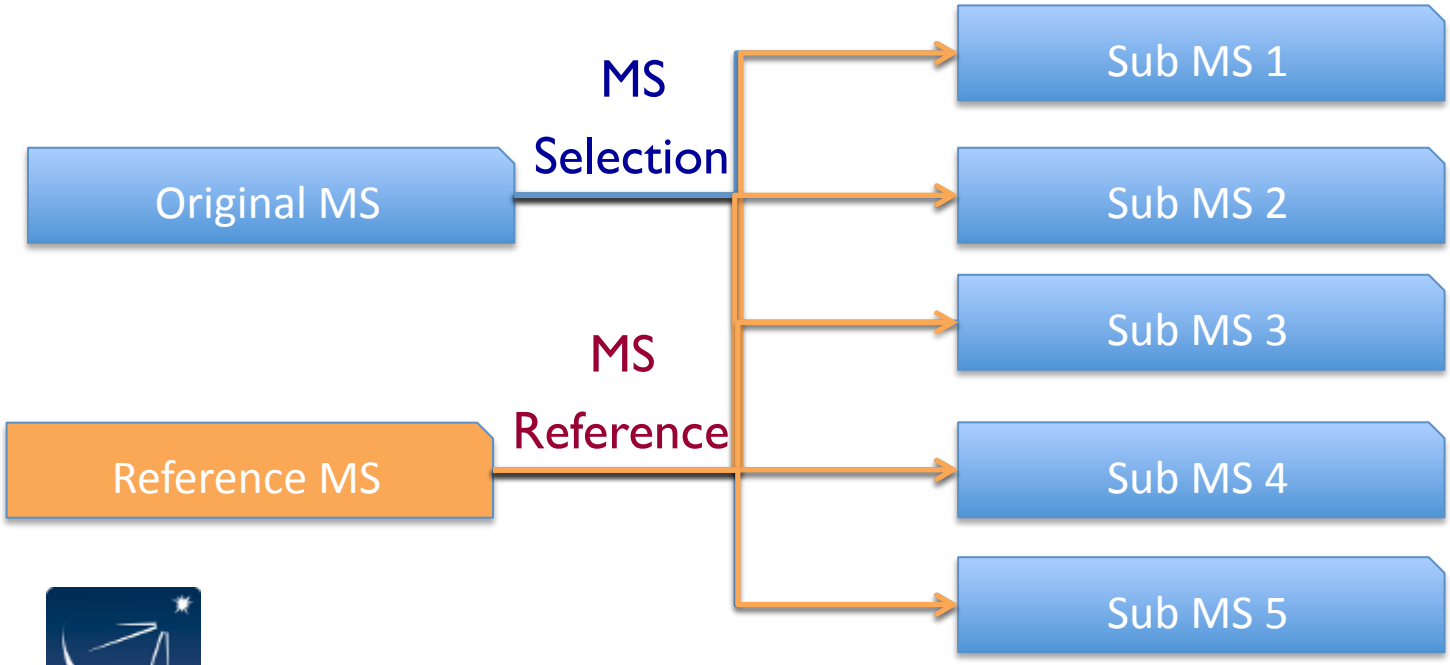
-Wikipedia

- Many of the tasks which require access to large amounts of data fit this description: Flagging, Time averaging, Applying calibration
  - Break the data into smaller pieces and distribute

- Imaging does not strictly fit this definition but the communication between processes is only at certain points in the cycle so it is also fairly easy to solve.
  - Break the problem into parts and distribute / gather as needed (major cycles are parallelized)
  - Use multi-threading for the gridding step

Reference MSs

- The easiest way to parallelize in CASA is to have multiple instances of CASA each running on a subset of the data.
  - To simplify the interface and minimize bookkeeping for the user we use a reference MS which is created via the partition task.



Parallelization of Pre-Imaging Tasks

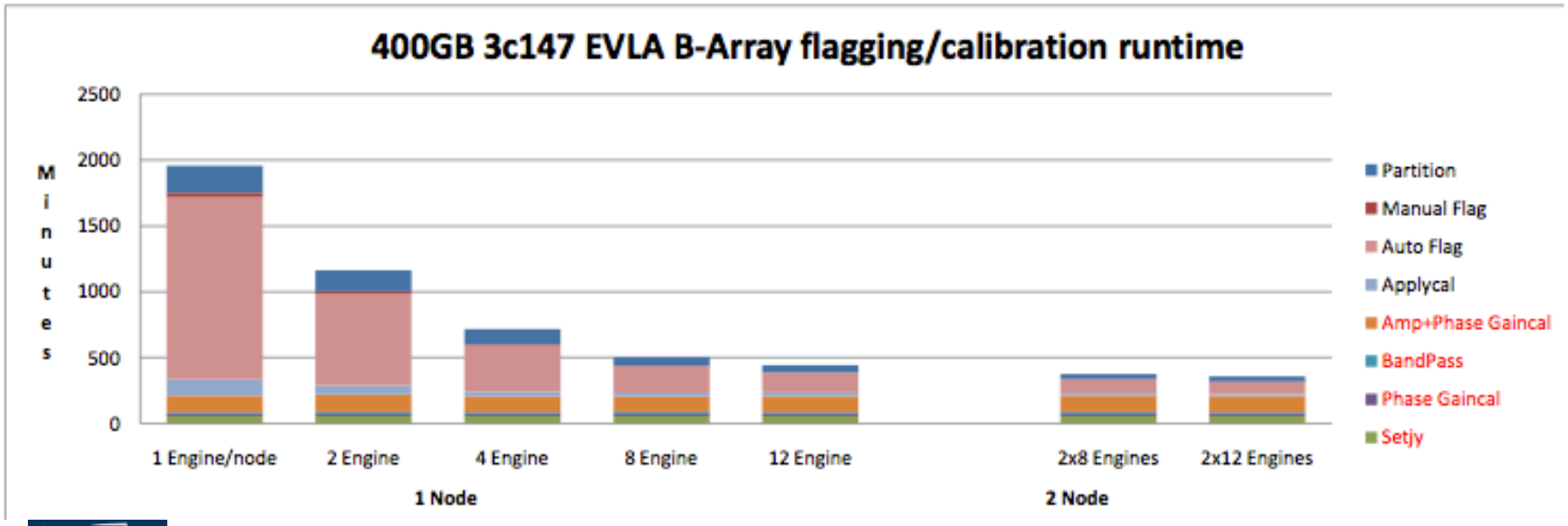
- After partitioning, tasks that work on the MS in place (e.g. don't create a new MS) and don't modify the subtables can be parallelized fairly easily.

- So far we've done those tasks which offered the most significant performance improvement for minimal investment :
  - Flagging (manual and auto)
  - Applycal
  - Setjy
  - Virtual Concat

- Another major effort in CASA 4.0 to make most tasks accept the multi-MS even if the task itself is not parallelized (minimize going back and forth)

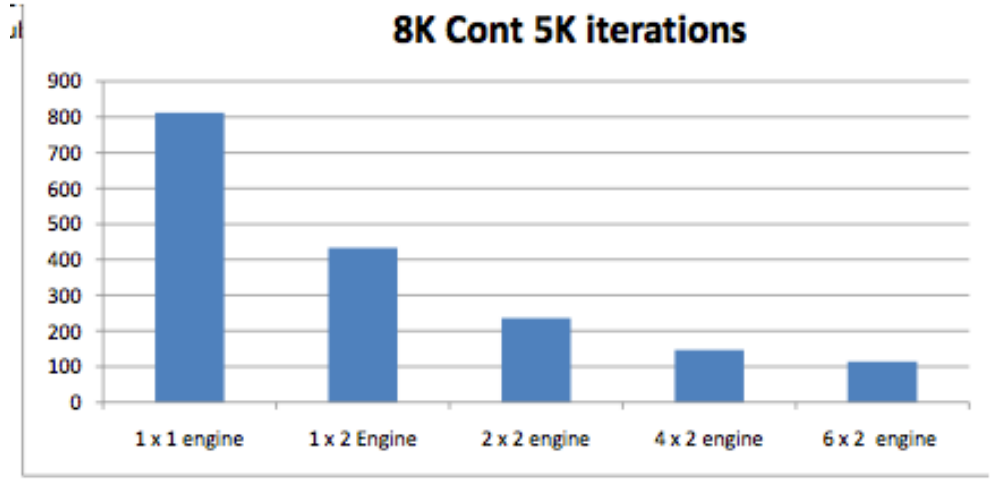
- This work is the major contribution by EU CASA staff (3) this cycle

	1 Node Total Engines/node					2 Nodes Total Engines/node	
	1 Engine/node	2 Engine	4 Engine	8 Engine	12 Engine	2x8 Engines	2x12 Engines
Partition	204	158	112	65	58	38	38
Manual Flag	36	18	9	5	3	3	2
Applycal	124	67	34	24	26	16	15
Auto Flag	1384	701	354	204	147	107	96
Setjy	55	56	55	55	55	55	55
Phase Gaincal	17	17	17	17	17	17	17
BandPass	13	14	13	14	13	14	13
Amp+Phase Gaincal	124	132	120	119	123	124	121
Parallel fraction	1748	944	509	298	234	164	151
Serial fraction	209	219	205	205	208	210	206
Total calibration	1957	1163	714	503	442	374	357
	1x 1 engines	1 x 2 engines	2 x 2 engines	4 x 2 engines			
P-continuum (8kx8k)	812	433	NA	NA	NA		
P-continuum (3kx3k)							
P-cube							



Cluster/Lustre & Parallel Clean

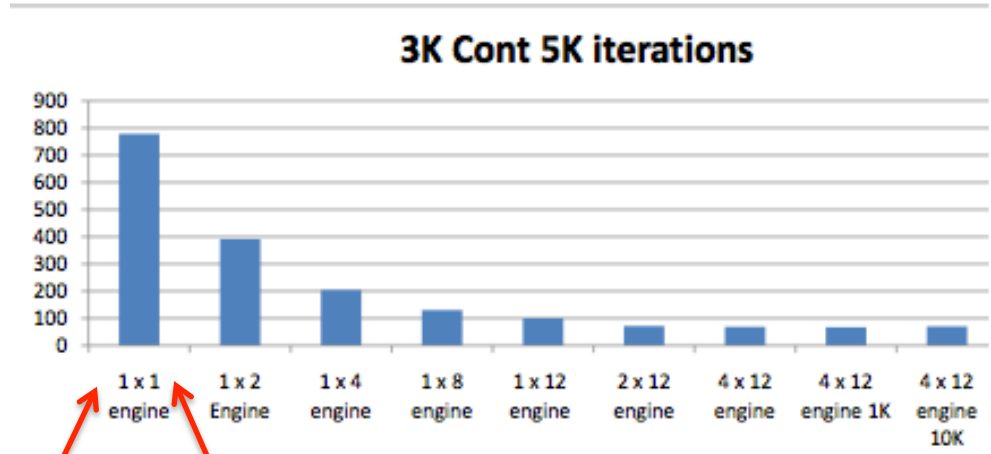
400 GB Continuum
- 8000x8000 pixel image
- 3000x3000 pixel image



Each core has 2 GB memory
With 8 cores, 16 GB available

For good performance the image and all its constituents (13 at the moment!) need to fit in memory!

8k image = 512 MB x 13 = 6.7 GB, thus 2 engines per node is max sensible



nodes # cores