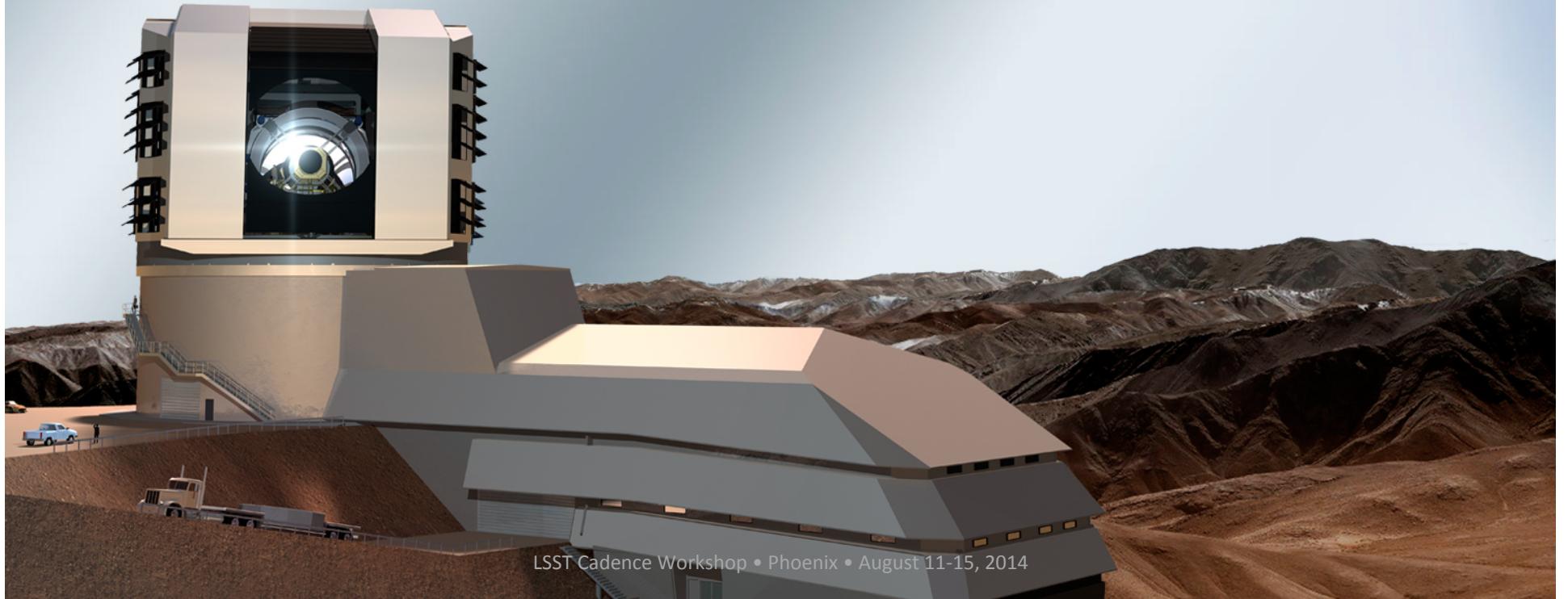




Metric Analysis Framework

Peter Yoachim
University of Washington





links



- Emails
 - Peter: yoachim@uw.edu
 - Lynne: ljones.uw@gmail.com
- Conference talks:
 - [https://github.com/LSST-nonproject/
UK_cadence_workshop_2015](https://github.com/LSST-nonproject/UK_cadence_workshop_2015)



A better name?



Survey Performance Analysis Tool SPAT

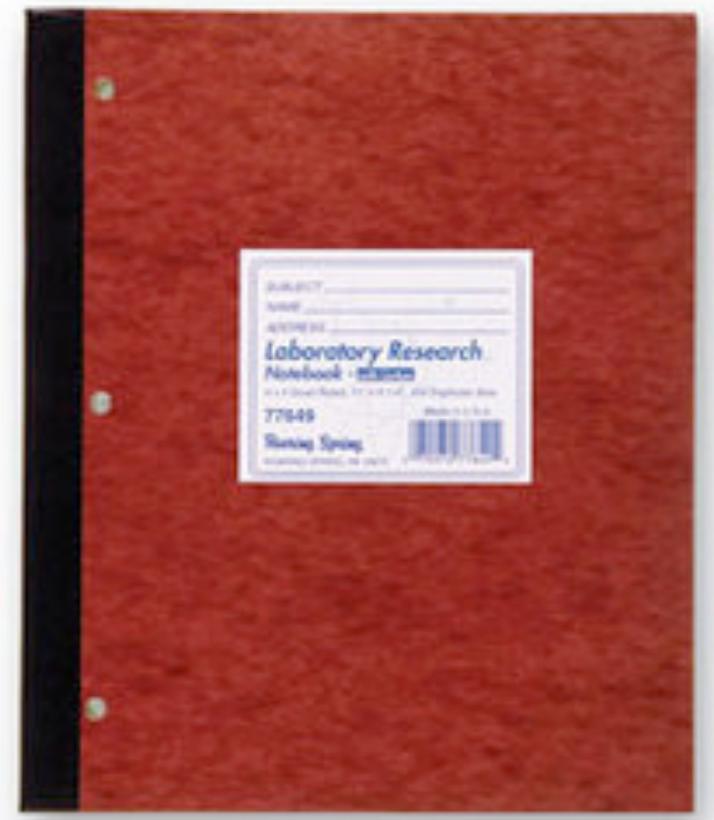




Classical Astronomy



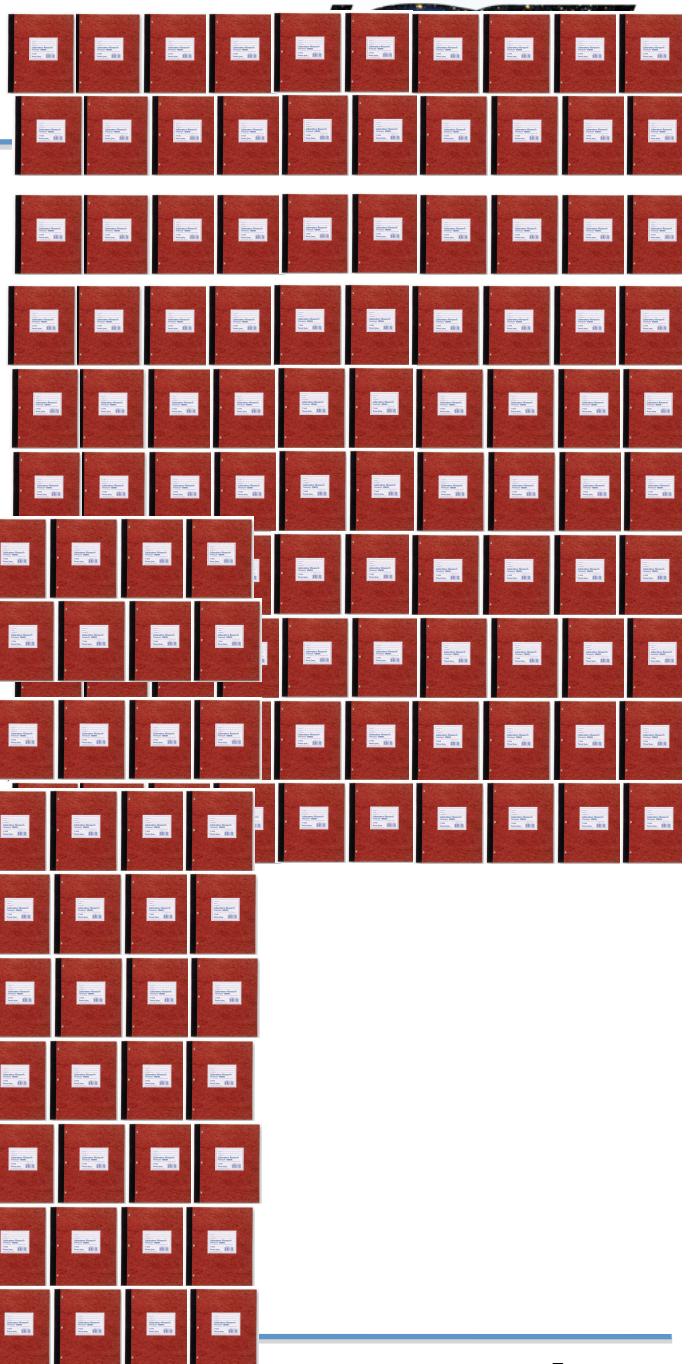
- You go to a telescope and take some data
- Did you get enough data to do your science?
 - Flip through the log book and check





LSST Era Astronomy

- Over 10 years, LSST will make around 2.5 million visits
- Need a tool to analyze ~300 log books





MAF Goals



- The goal of the Metric Analysis Framework is to provide an easy way to visualize the properties of a survey and quantify the science that can be done with that survey
- Able to run in an automated fashion so we can compare large numbers of simulated surveys
- Easily extended so users can contribute their own analysis—we want YOU to show us the best way to measure a survey's performance



- Basic info about MAF
 - A framework for finding out how well an executed survey meets your science needs.
 - All Python, part of the LSST stack
 - Install instructions here:
<https://confluence.lsstcorp.org/display/SIM/Catalogs+and+MAF>

General problem of data analysis and dimensionality reduction
2.5 million observations → data cube → sky map → single number



The Operations Simulator



- OpSim is our source for what these 2.5 million visits might look like.
 - Includes real weather log
 - Scheduled and unscheduled down time
 - A scheduler that balances several science goals
- OpSim scheduler based on “Proposals”
 - Wide-Fast-Deep (aka, “the main survey”): Cover ~20,000 sq deg
 - North Ecliptic Spur: Solar system objects
 - Deep Drilling Fields: ~6 deep fields
 - Galactic Plane
 - South Celestial Pole



Opsim Output



- An OpSim database (distributed to the community as sqlite files) contains multiple tables that record the observations taken, state of the telescope, configuration of the scheduler, etc.
- For our interests, we will focus on the Summary table in each OpSim run database. This is your very detailed observing log book.
 - Includes all your favorite values
 - FieldRA, FieldDec, airmass, expMJD, filter
 - More detailed things as well: Slewtime, skybrightness, point source limiting magnitudes

~30 total columns in Summary table. See:

<http://ls.st/5d8>

For more documentation on the Summary table



OpSim Warnings



- OpSim records LSST “visits”
 - 1 visit = 2 back-to-back exposures!
- By default, OpSim uses fixed fields and does not dither
- There can be “repeated” entries in the Summary table if a visit is used for multiple proposals (e.g., a visit can count towards WFD and Deep Drilling).

In MAF, we take care of this by doing a SQL “group by” on the expMJD.

Observation for	Object	MJD	RA	DEC
Jen	M31	3456.4	0:42:44	41:16:09
Jen	Feige 34	3456.8	10:39:37	43:06:09
Dave	Feige 34	3456.8	10:39:37	43:06:09
Dave	M1	3457.1	05:34:32	22:00:52

Not unique

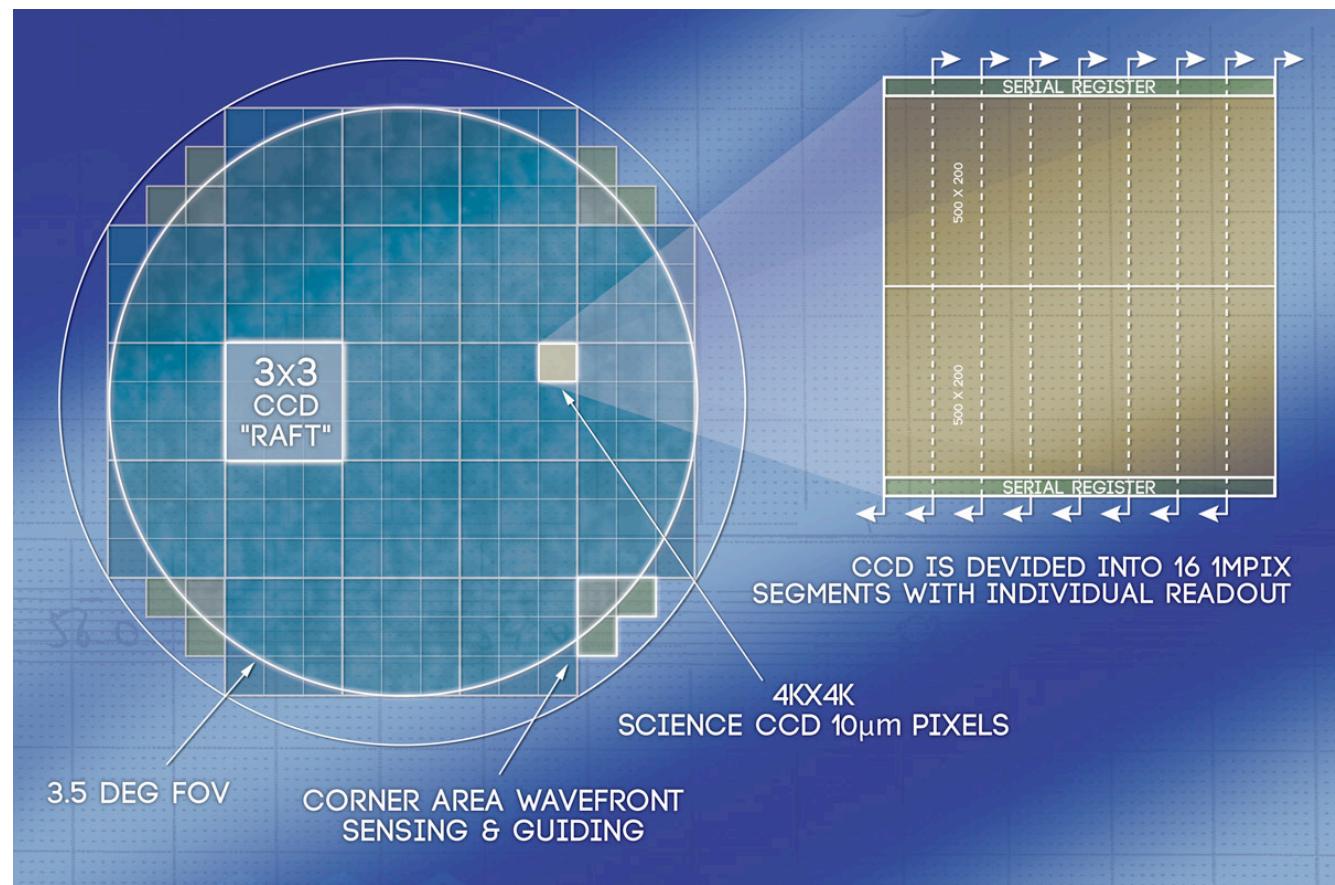


MAF Warnings



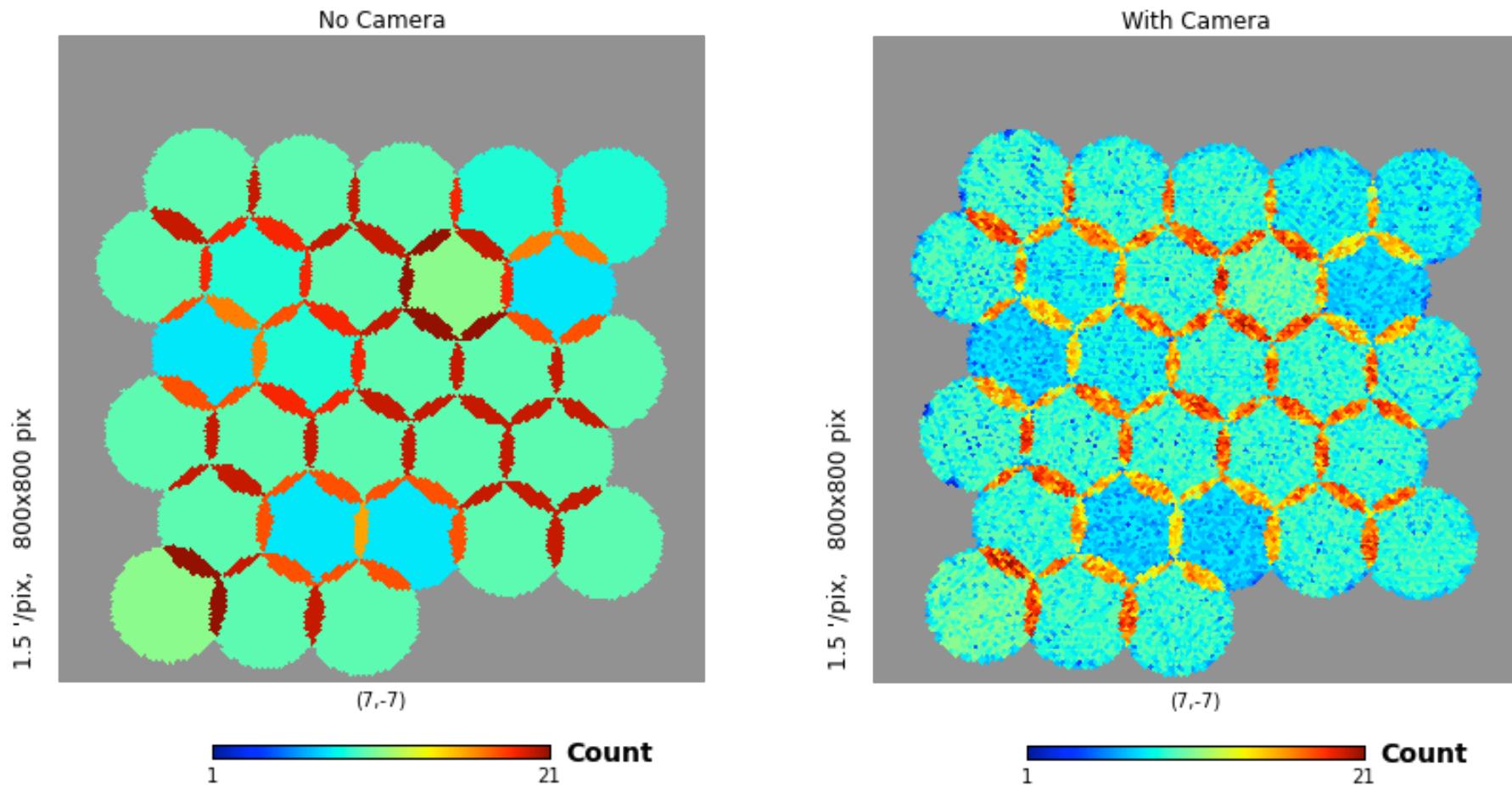
- By default, we assume if an object is within 1.75 degrees of the center, it gets observed.
- This is probably a few % optimistic given chip gaps, etc.

You can run with the full focal-plane geometry, but it slows things down a bit.





With and Without Chip Gaps





How do we quantify how well a survey performs?



- Some things we commonly ask
 - What's the median seeing of all the visits?
 - How many exposures do we get per night?
 - What's the co-added depth of a deep drilling field?
 - What's the average co-added depth at the end of the survey?



To answer these we have to divide the observations up



- Some things we commonly ask
 - What's the median seeing **of all the visits?** **All the observations**
 - How many exposures do we **get per night?** **Observations per night**
 - What's the co-added depth of a **deep drilling field?** **Observations at a point in the sky**
 - What's the average co-added depth at the end of the survey? **Observations at many points in the sky**

In MAF, splitting the visit up correctly is done by “**slicers**”



To answer these we have to compute some algorithm



- Some things we commonly ask
 - What's the *median* seeing of all the visits?
 - *How many* exposures do we get per night?
 - What's the *co-added depth* of a deep drilling field?
 - What's the *average co-added depth* at the end of the survey?

In MAF, computing algorithms is done by “metrics”

To answer these questions with MAF, you take observations from a database and then pair a slicer and a metric.



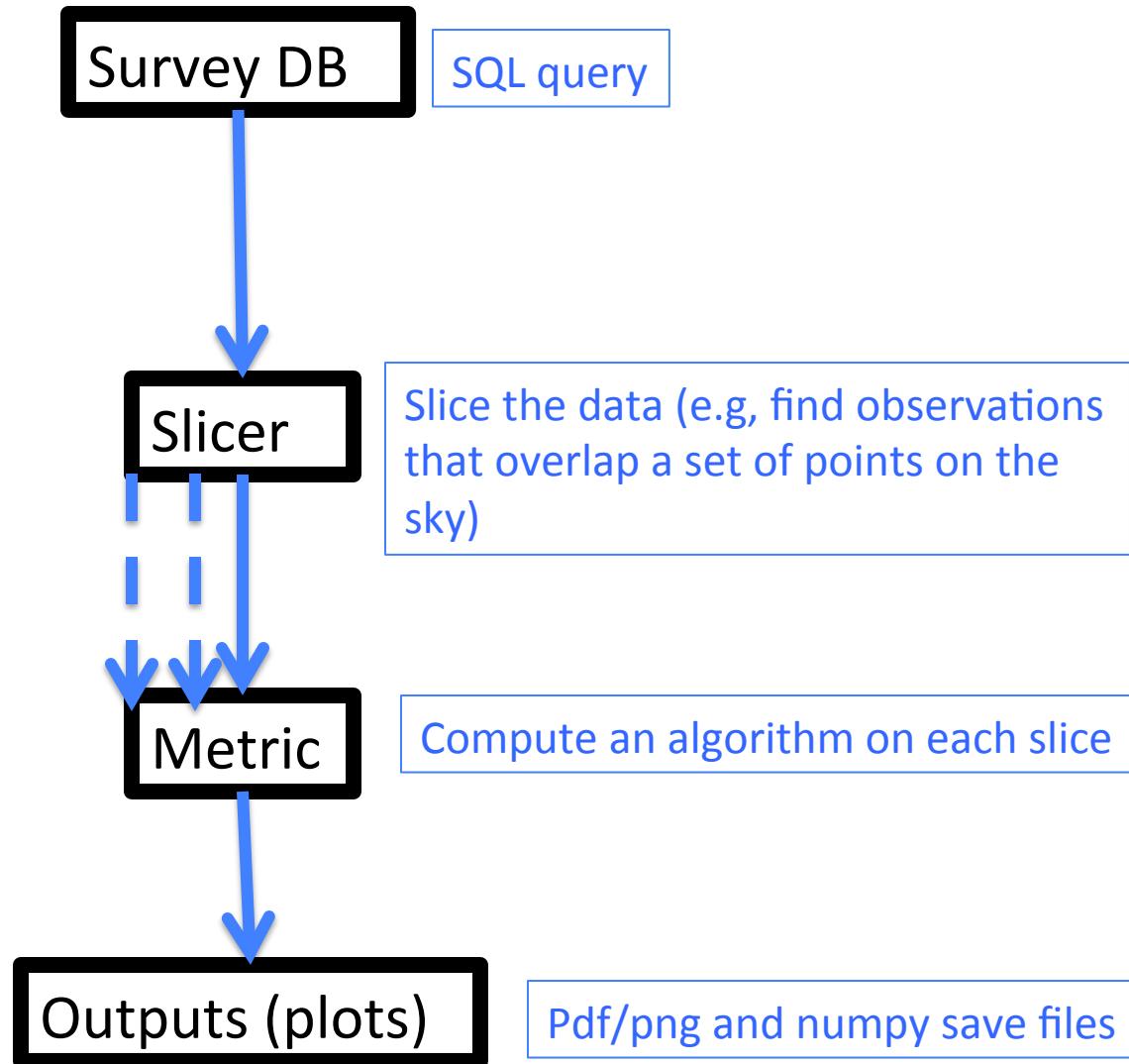
The Three Primary Slicers



- UniSlicer
 - Passes all data directly to the metric algorithm.
 - How to answer, “what’s the mean seeing?”, “What is median airmass?”, etc.
- OneDSlicer
 - Slice data based on one parameter.
 - How to answer, “what’s the distribution of exposures **per night?**”, “What’s the distribution of slew times?”
- HealpixSlicer
 - Loops over points on the sky and returns overlapping exposures
 - How to get a map of the sky. Calculate co-added depths.

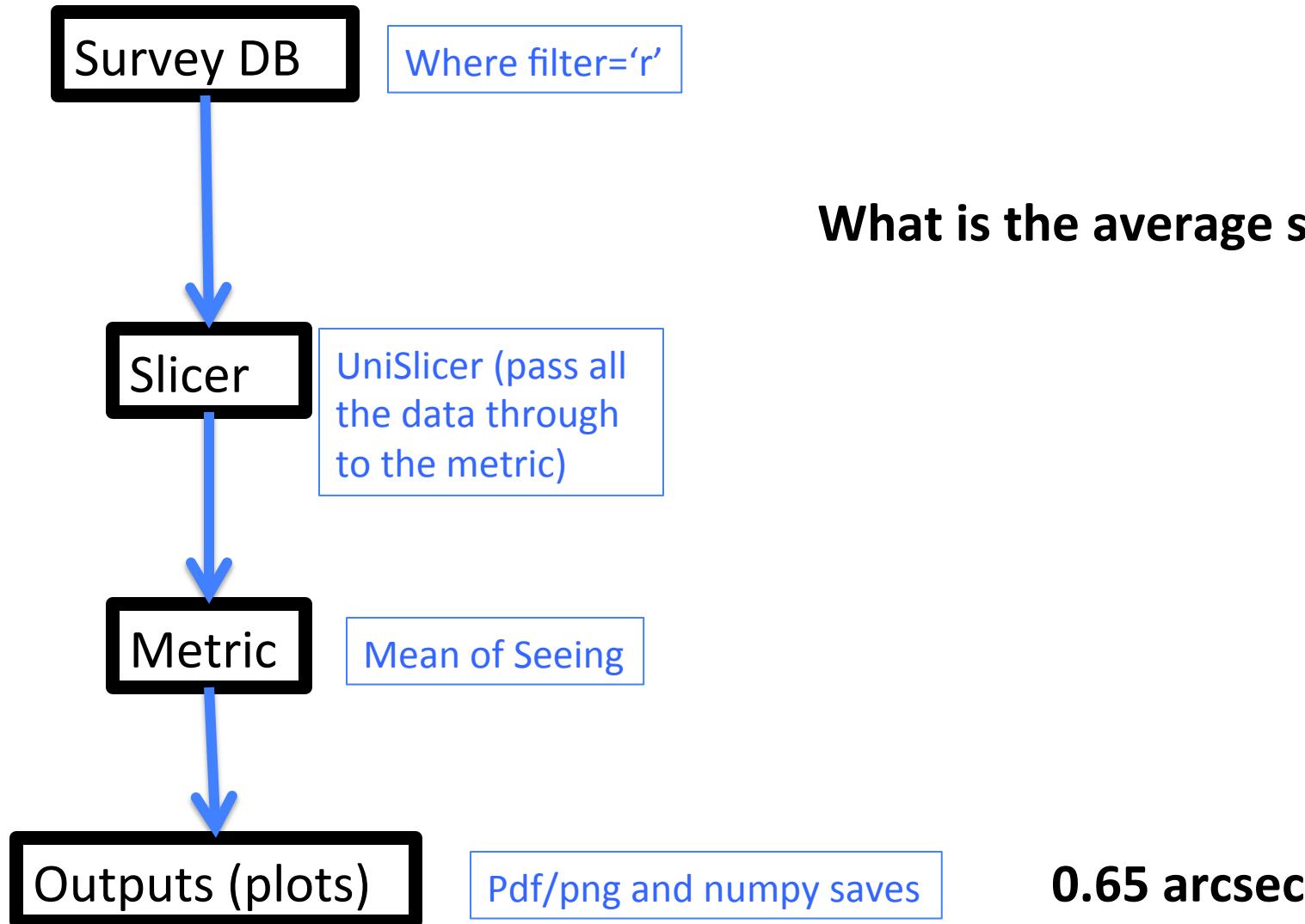


General MAF Outline





Example slicer + metric combinations

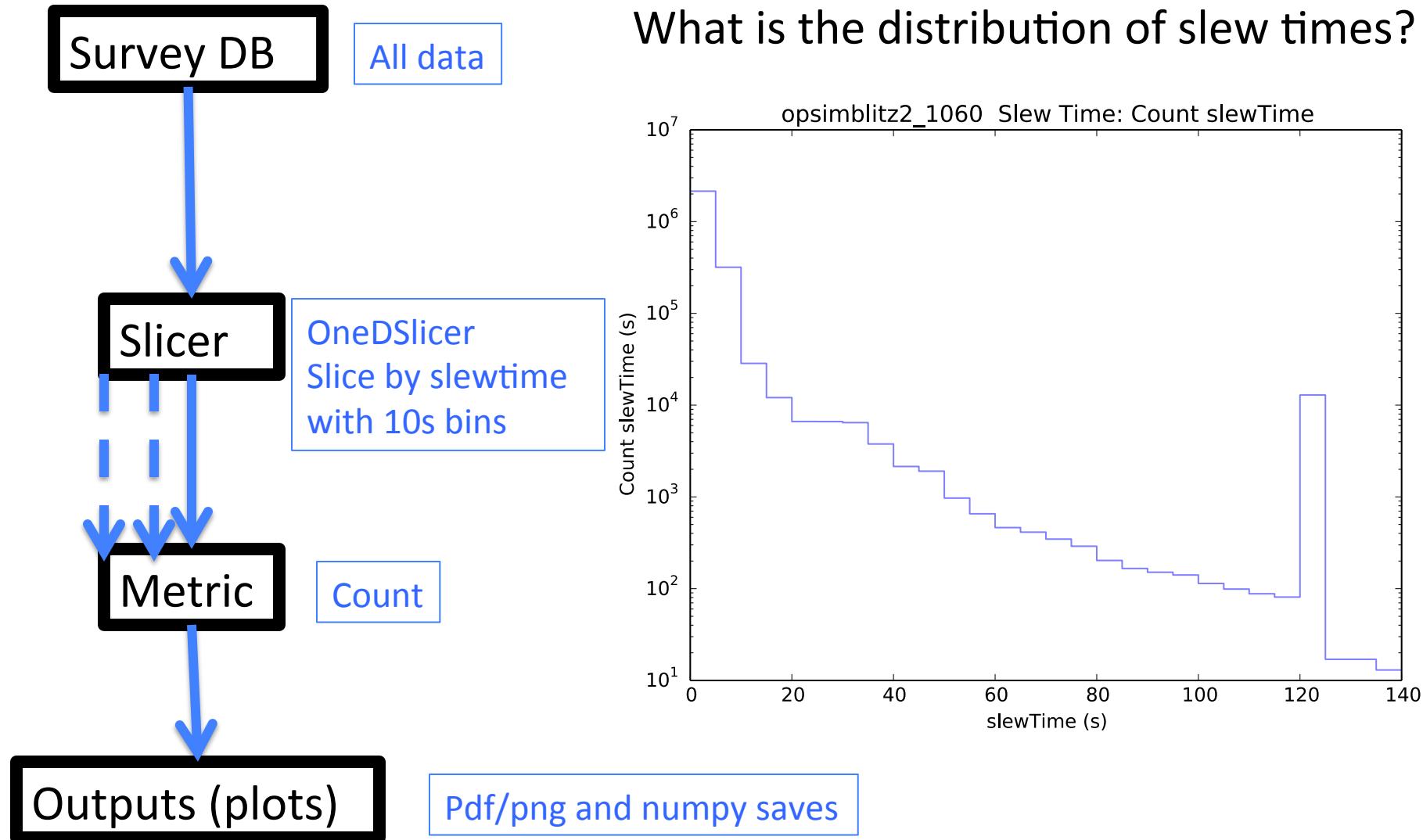


What is the average seeing in r?

0.65 arcsec

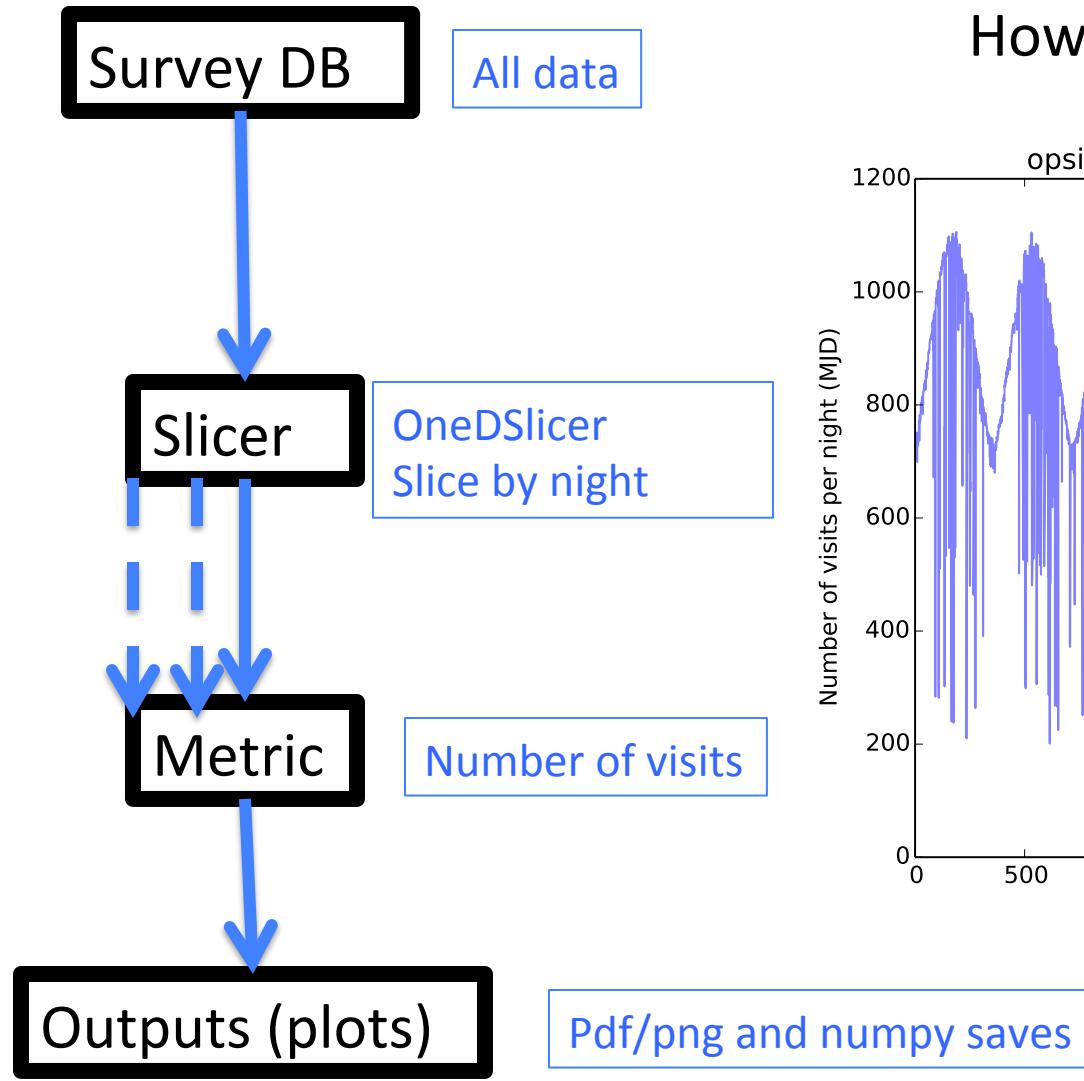


Example slicer + metric combinations

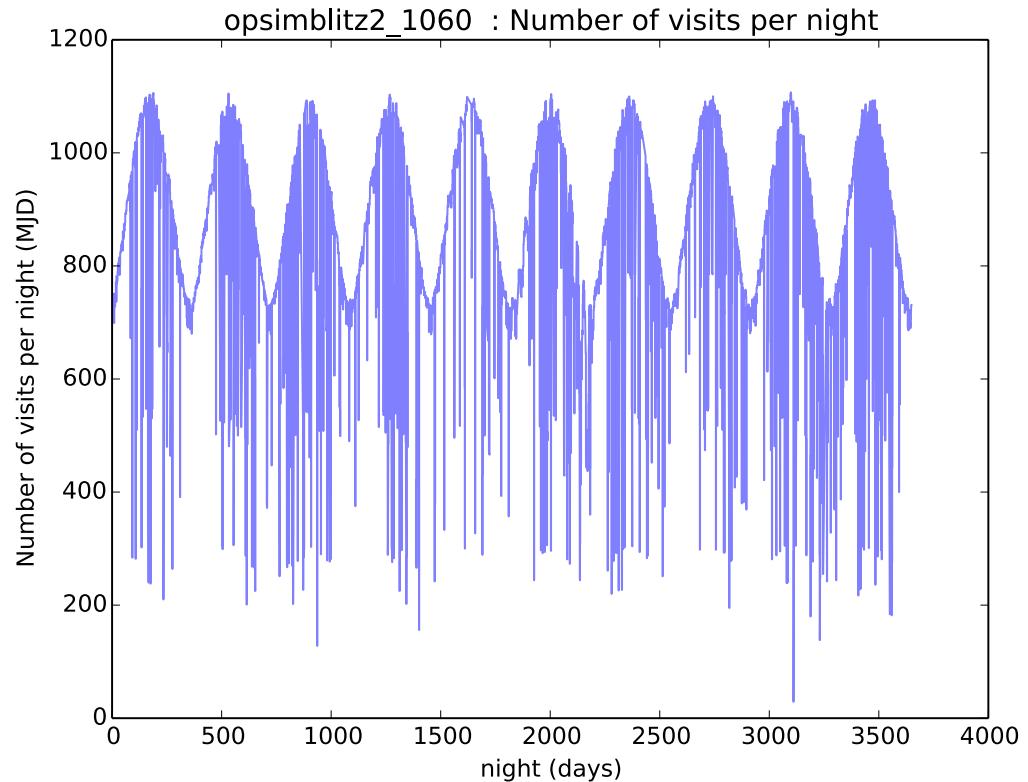




Example slicer + metric combinations



How many visits per night?



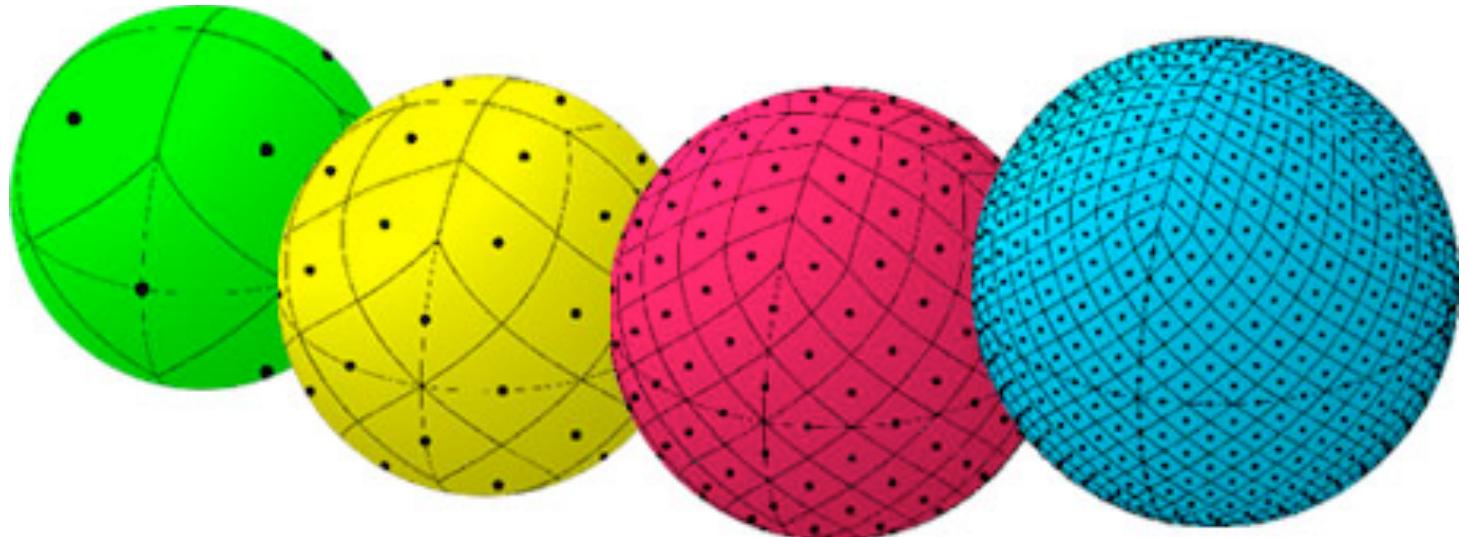
Seasonal oscillation and nights with weather and engineering downtime



HEALpixels



- For our spatial binning, we use HEALpixels
 - Equal area pixels
 - Popular with cosmologists, rapidly compute power spectra
 - Or, use a custom list of RA,Dec points
 - Can also bin by OpSim field ID. Faster, but field overlaps are not resolved





MAF Examples



Survey DB

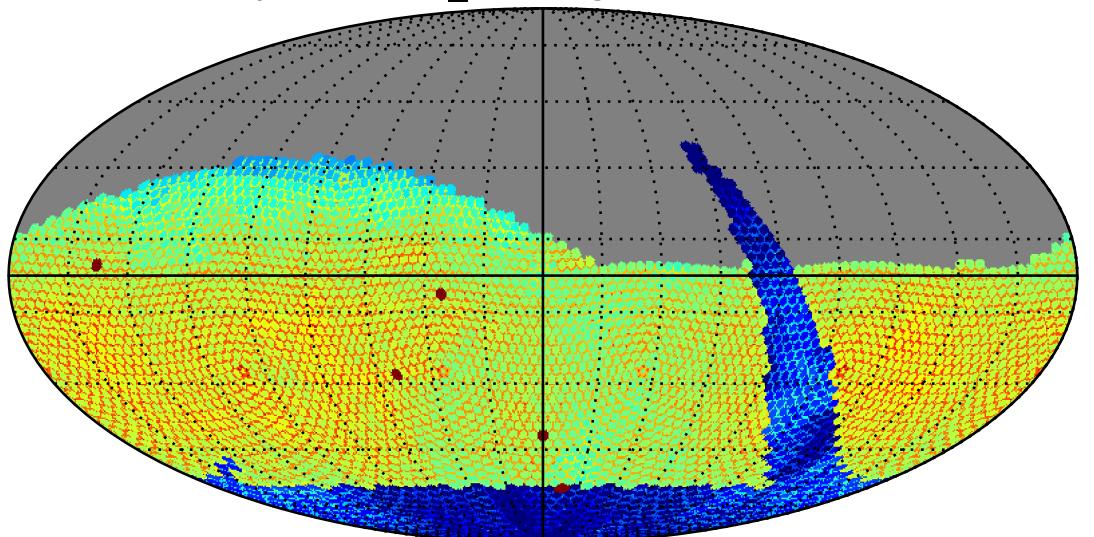
Filter = 'g'

What is the coadded depth across the sky in g band?

opsimblitz2_1060 g: CoaddedM5

Slicer

HealpixSlicer



Metric

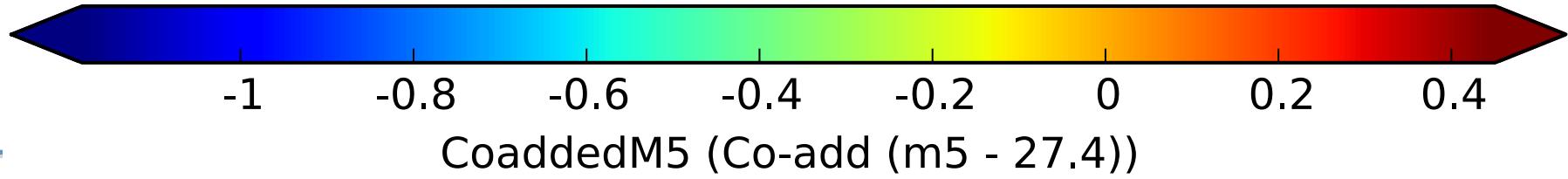
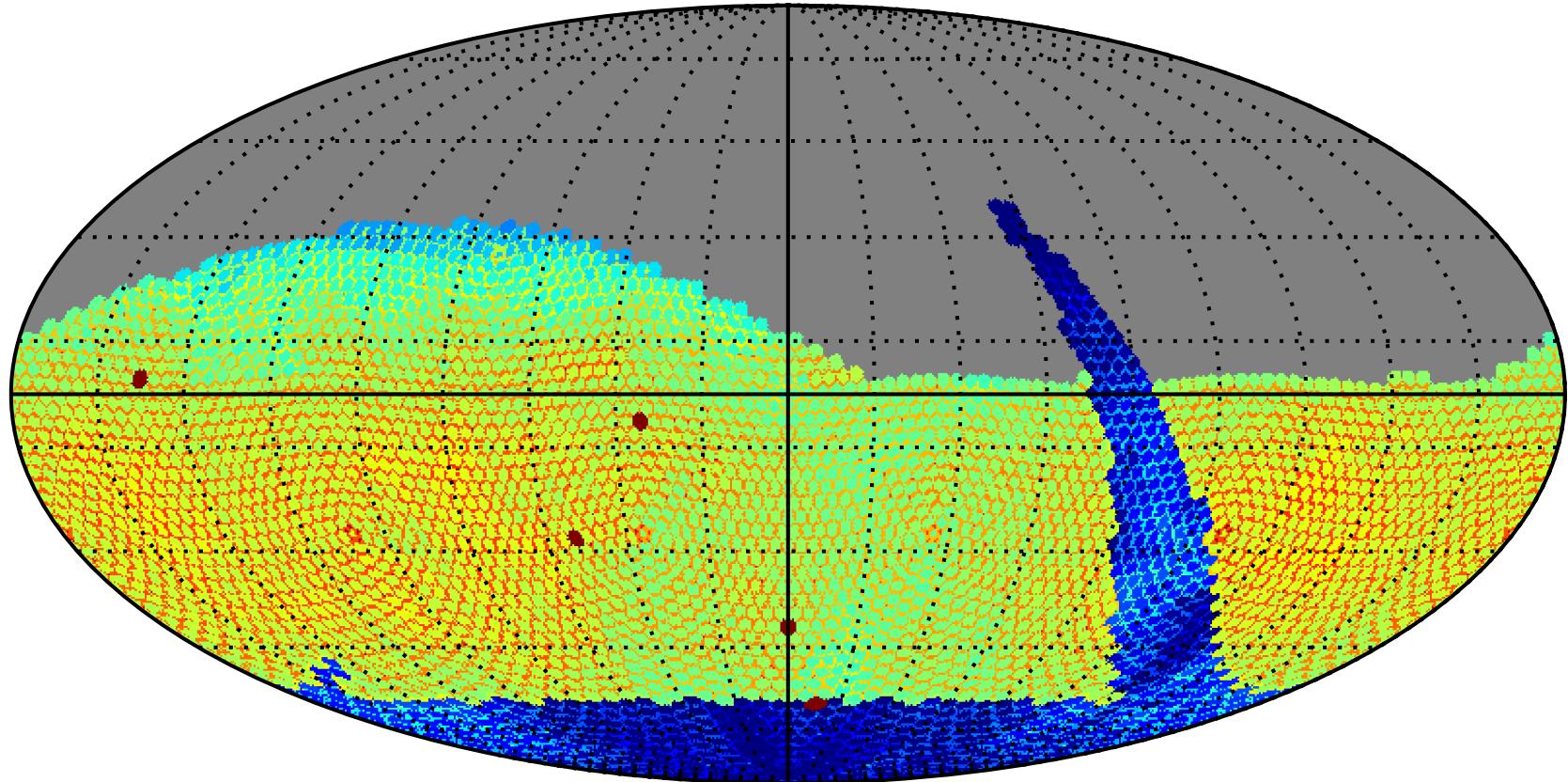
Coadded depth
Given single visit depths

Outputs (plots)

-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4

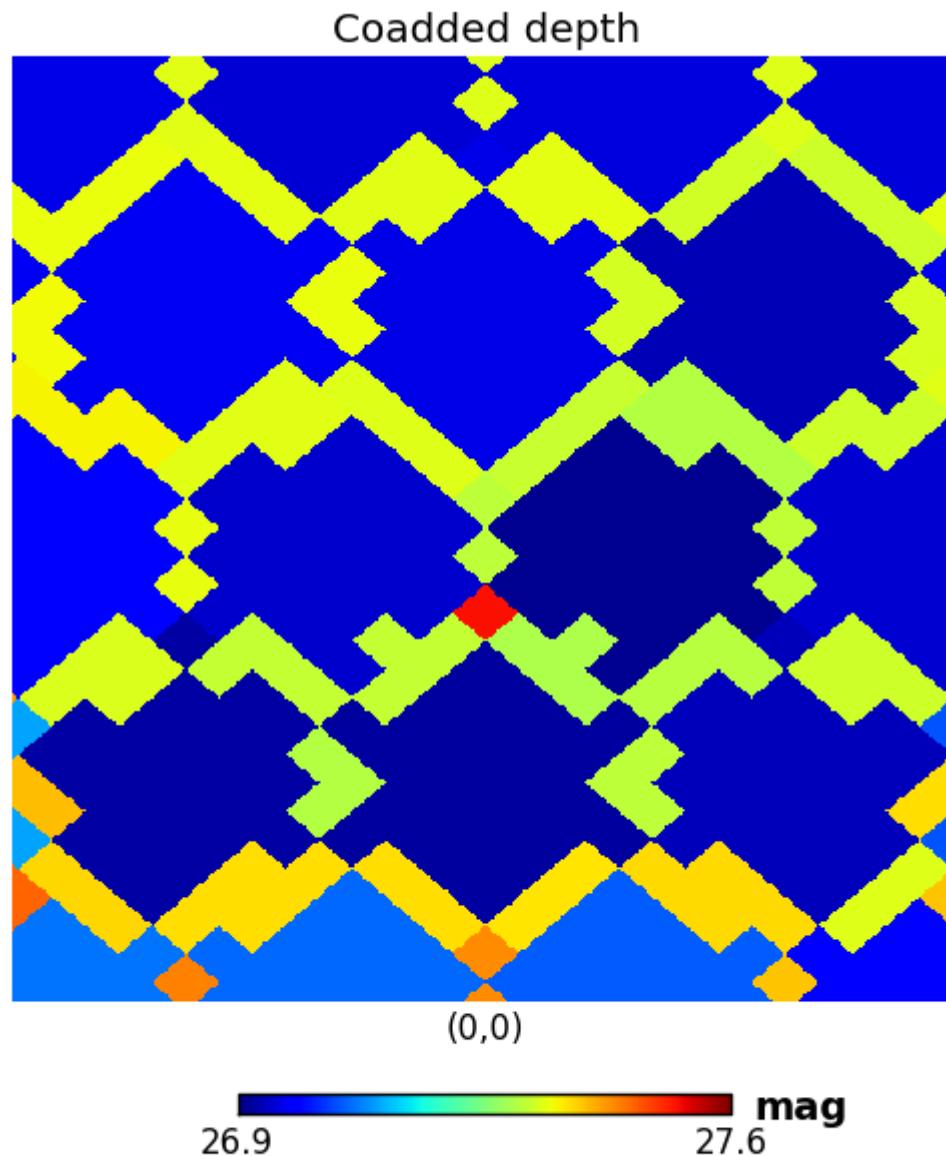
CoaddedM5 (Co-add (m5 - 27.4))

opsimblitz2_1060 g: CoaddedM5



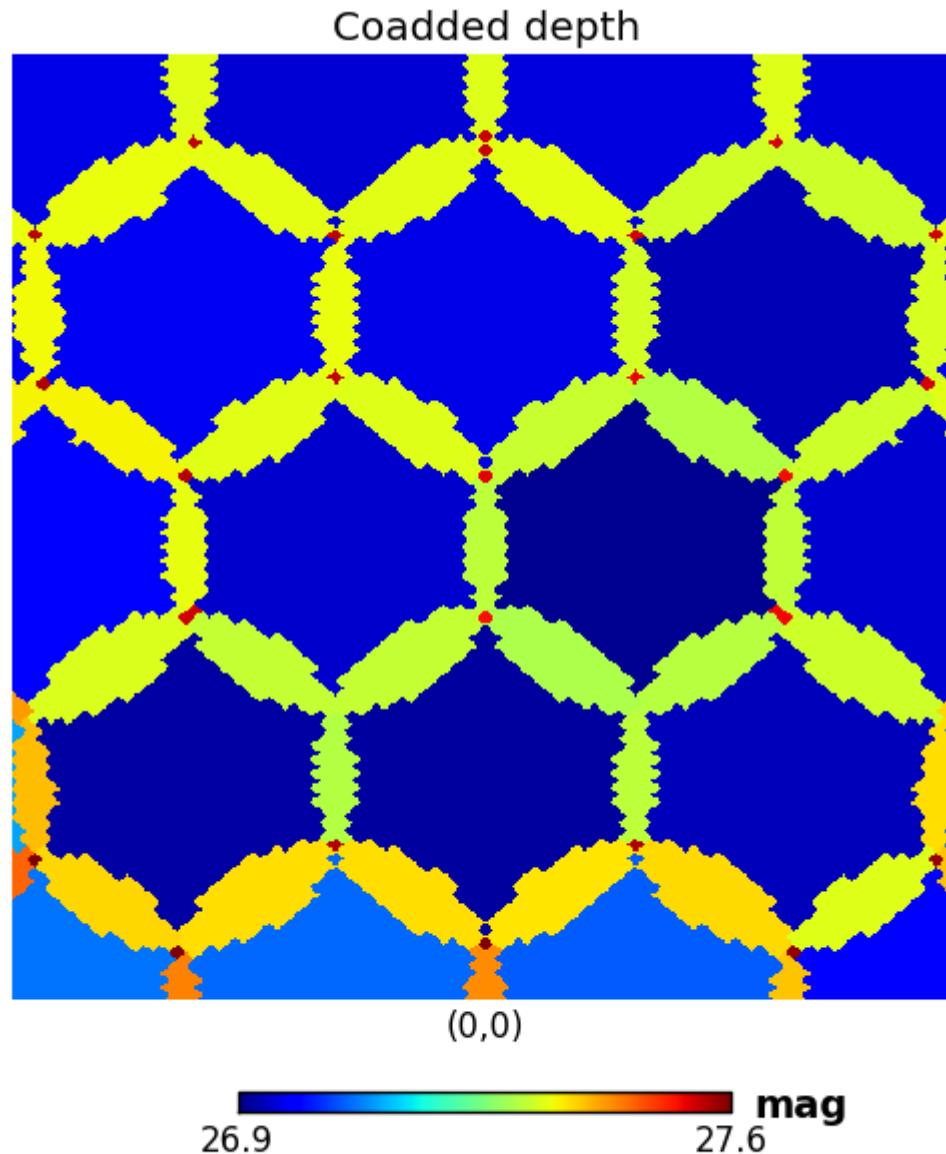


nside=128, 27.5 arcmin resolution



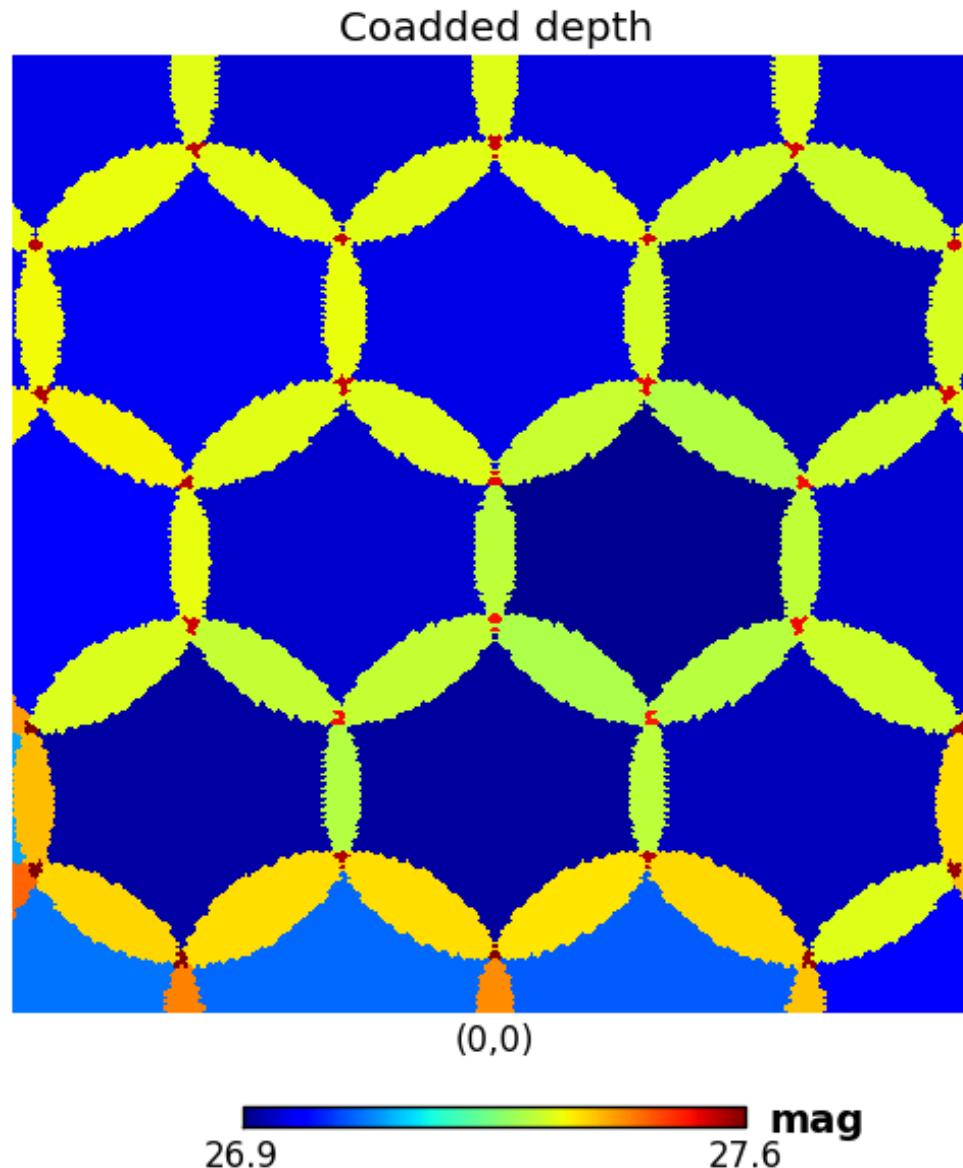


Nside=512, 6.9 arcmin resolution



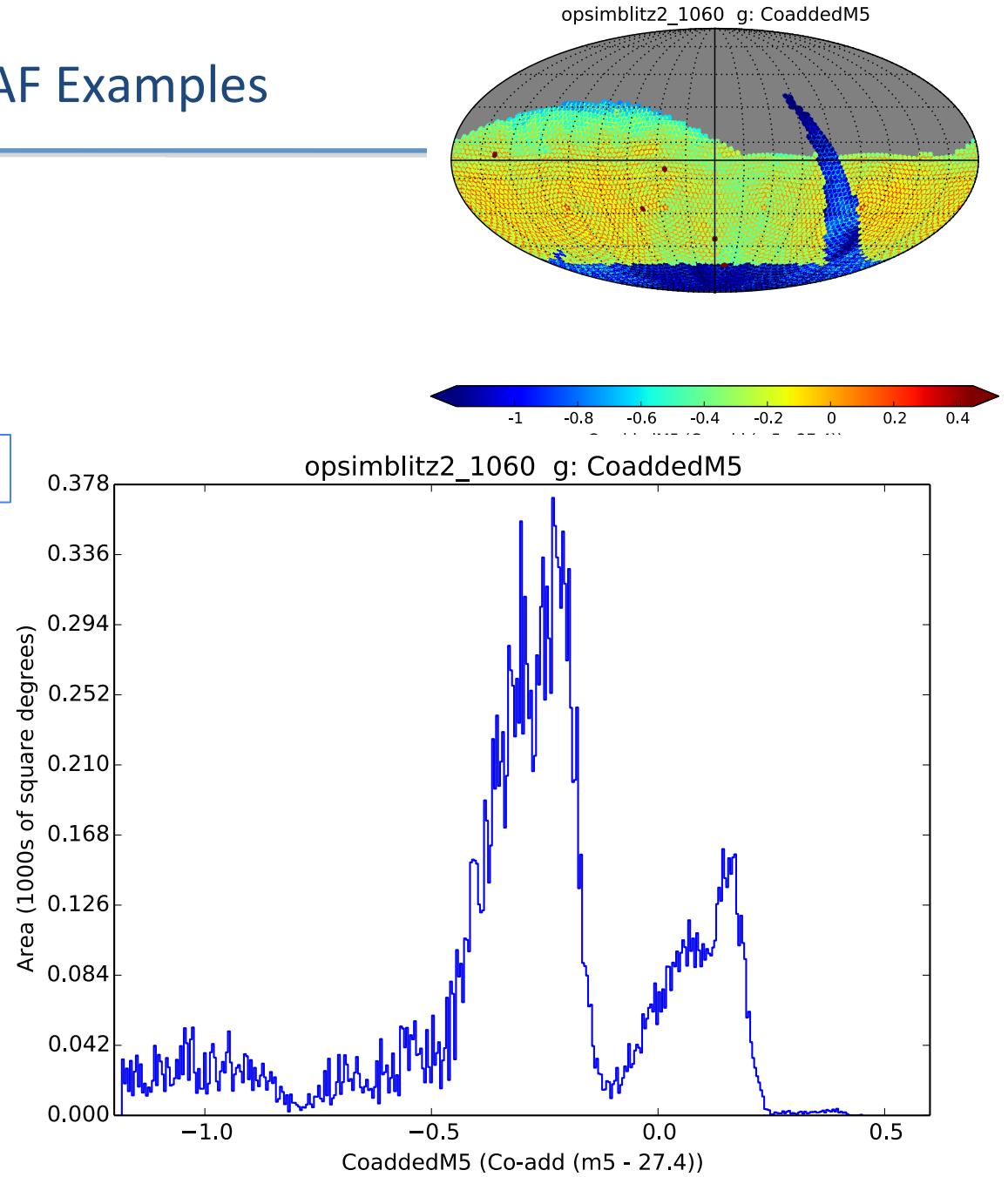
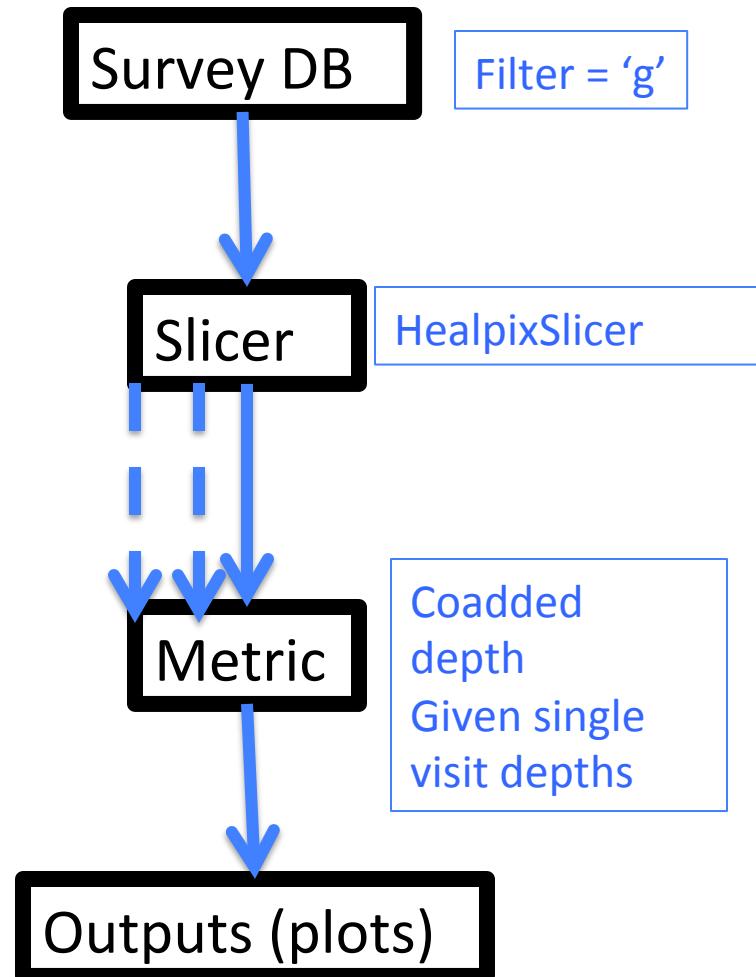


Nside=1024, 3.4 arcmin resolution



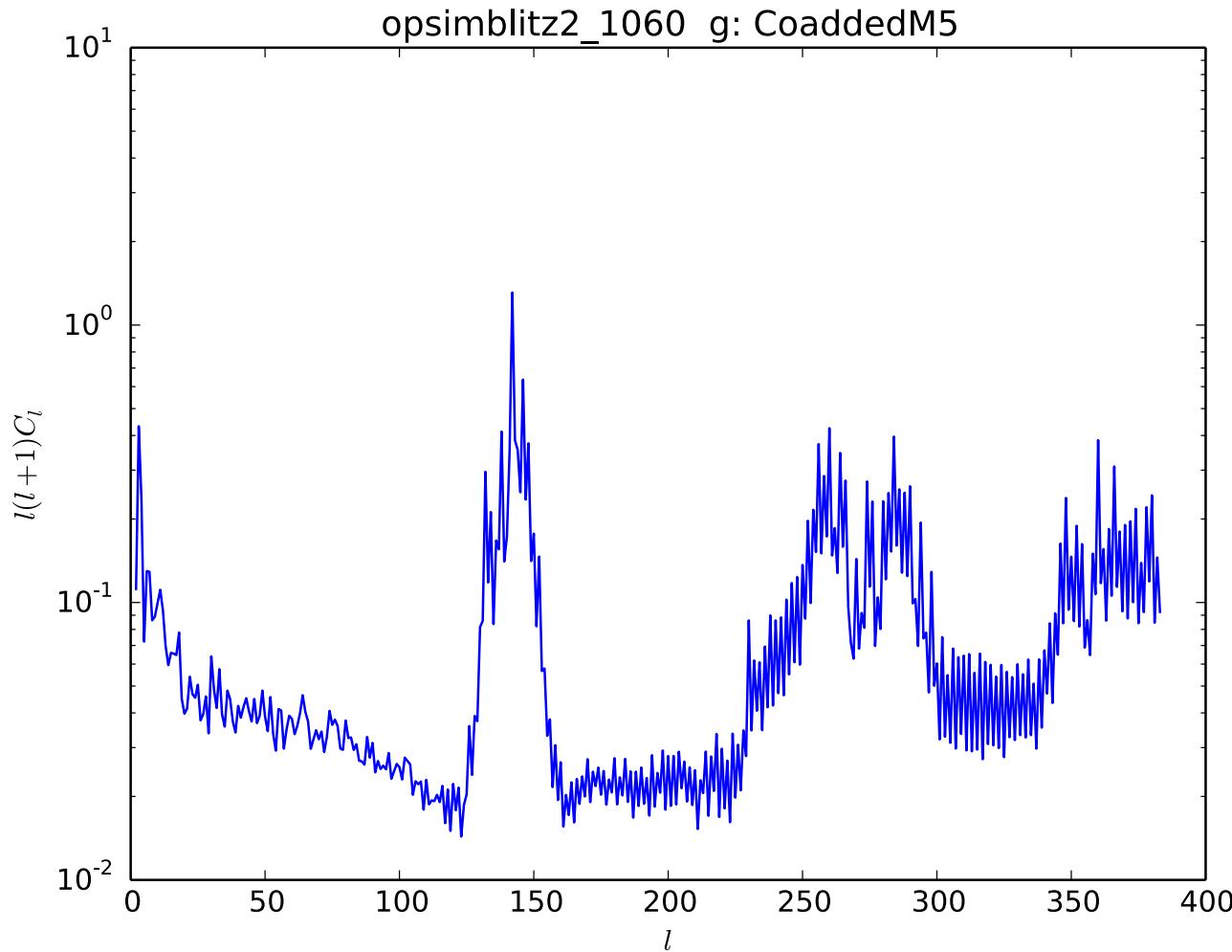


MAF Examples





Healpixels let us compute Power Spectra



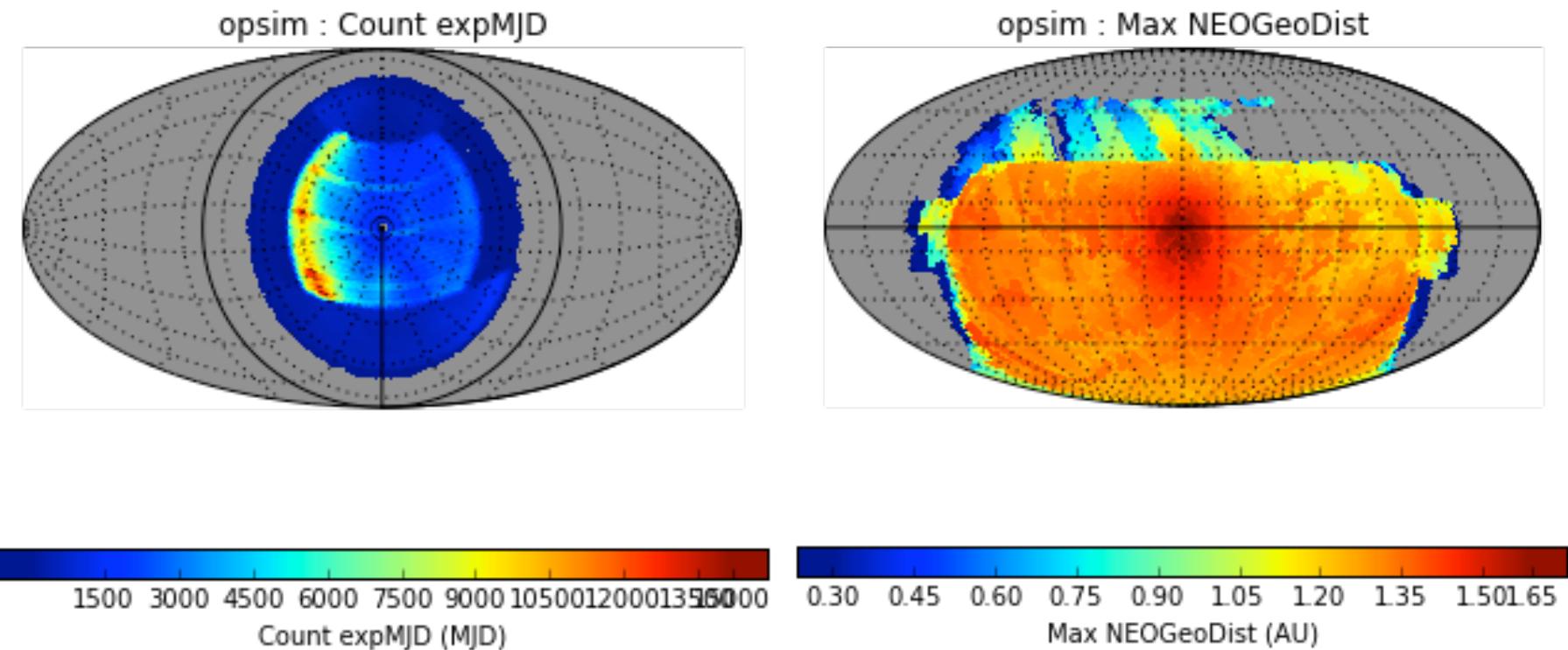
*Peak at $l=140$ (~ 1 degree),
caused by field overlaps
Peak at $l=270$ from triple
overlaps.*



Healpix beyond RA,Dec

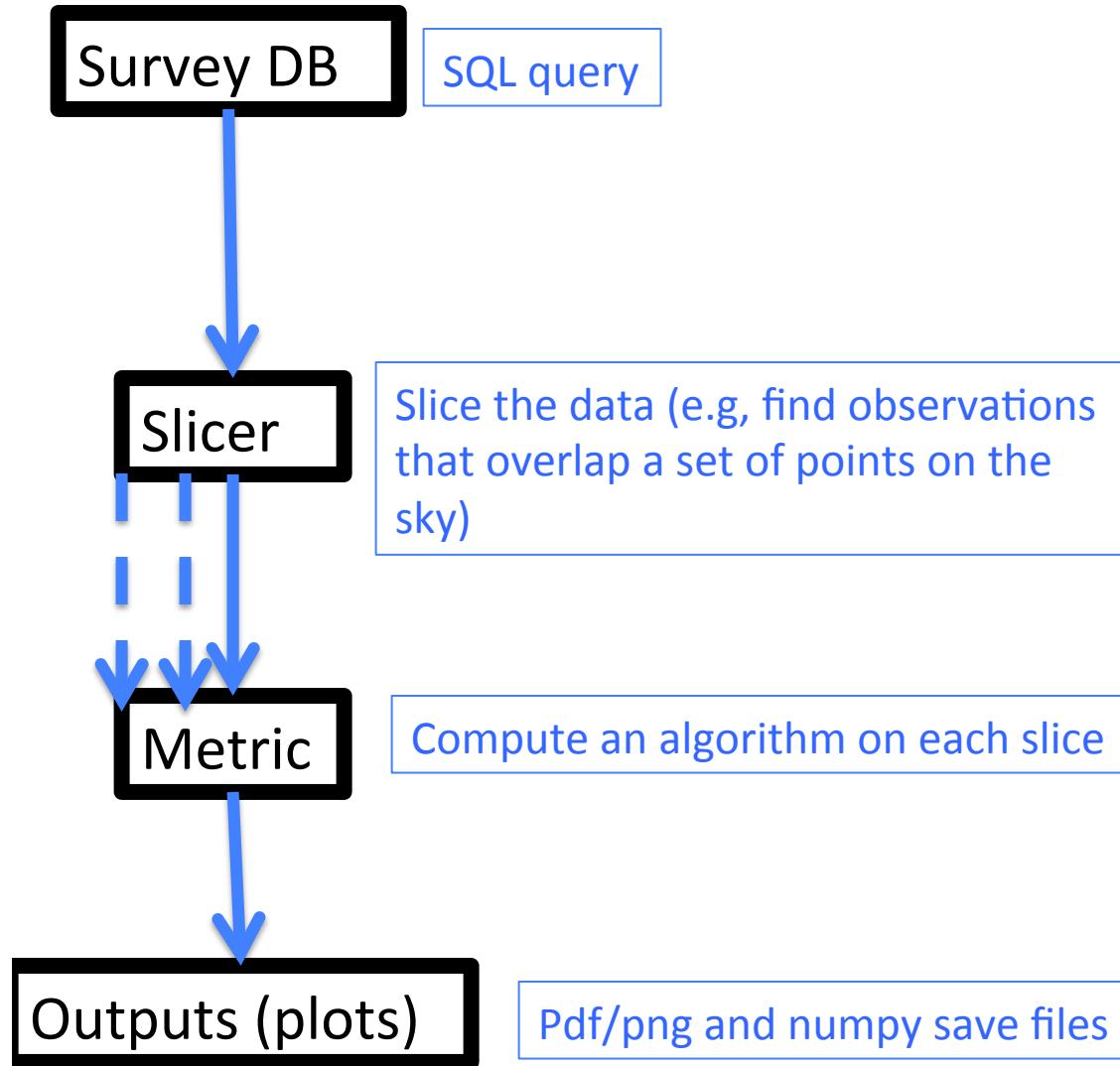


- The HealpixSlicer defaults to RA,Dec, but can be set to use Alt,Az or ecliptic coordinates.





Does it go in the SQL or the Slicer?



The slicer determines the type of output plot. We tend not to plot things as a function of filter or proposal ID, so those are set with the SQL where-clause

Sometimes we use SQL for time, e.g., if we want to look at the first year performance of the survey. Other times the slicer will slice on time.



Recap: slicer + metric



- Which slicer you pick will determine your output plots
- Three main slicers
 - UniSlicer: Pass all the data, returns a number
 - OneDSlicer: Slice by one variable (like per night), returns a single plot (like histogram)
 - HealpixSlicer: Slice on observations overlapping healpixels, return a skymap, histogram, and powerspectrum
- Many metrics to pick from
 - coaddm5
 - Max
 - Mean
 - Median
 - Min
 - FullRange
 - Rms
 - Sum
 - RobustRMS
 - Binary
 - FracAbove
 - FracBelow
 - OpenShutter
 - Completeness
 - Parallax
 - ProperMotion
 - Fft
 - Supernova
 - Uniformity
 - TemplateExists
 - VisitGroups



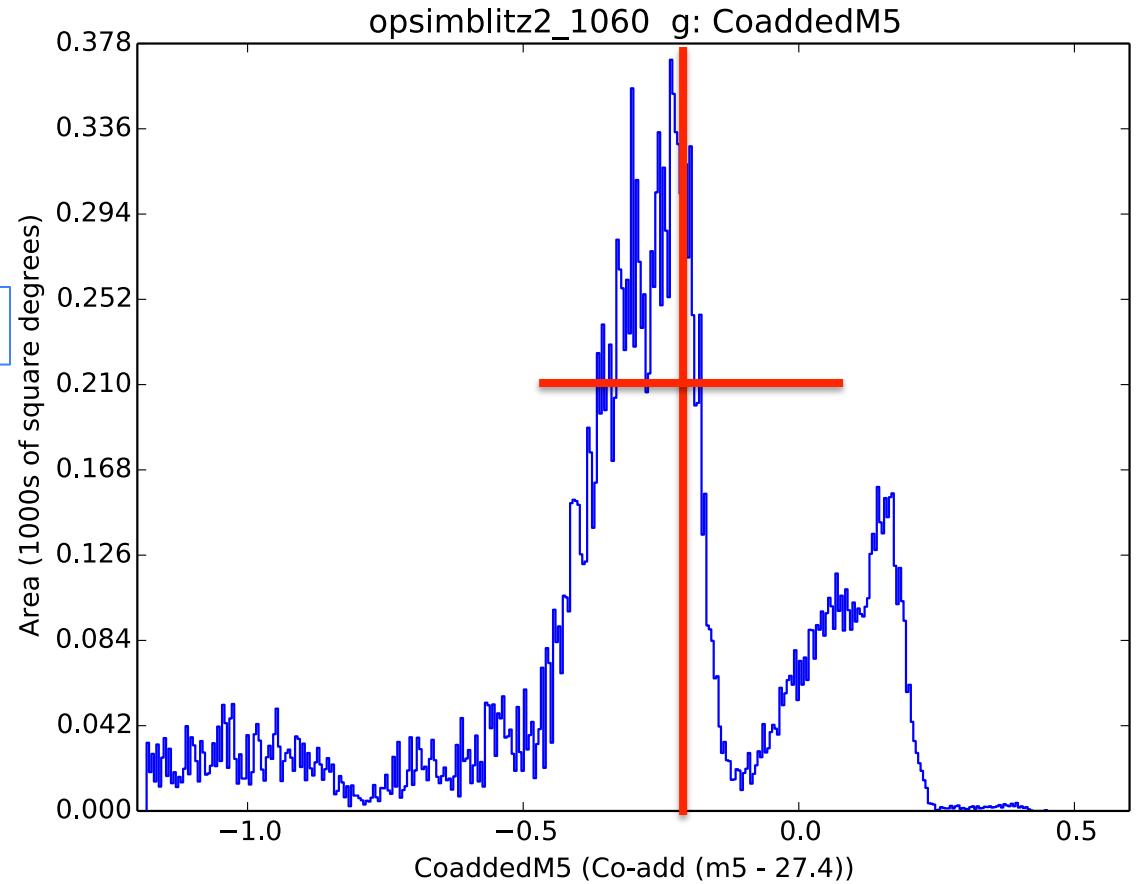
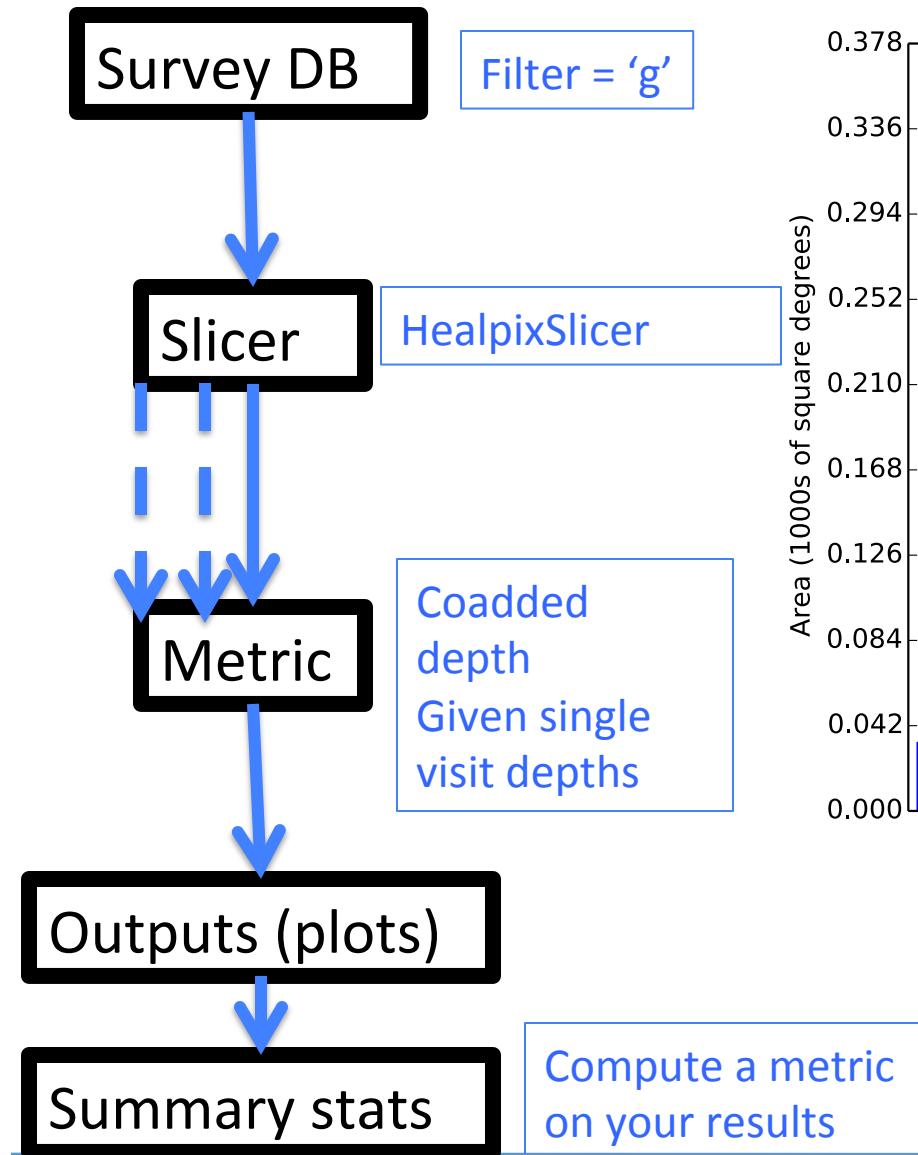
Additional Capabilities



- Summary Statistics
- Stackers: Adding derived columns

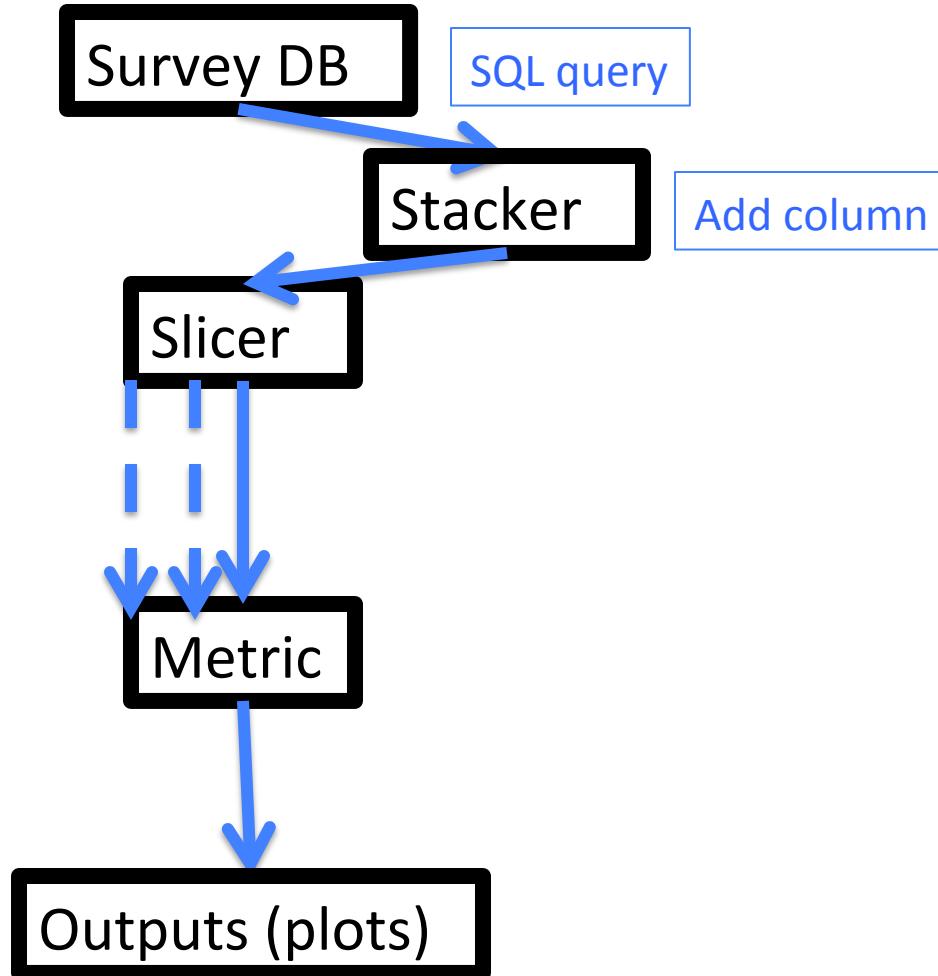


Summary Stats on a HealpixSlicer





Add a psudo-column



Sometimes it's convenient to calculate a new column that isn't present in Opsim before slicing the data.

For example, normalized airmass, parallax factor, dither offset, hour angle.

We have “stackers” that allow you to stack new columns onto the SQL results before they are passed to the slicer. (we’re calling them stackers since we are using the numpy stack functions to add columns to numpy arrays)



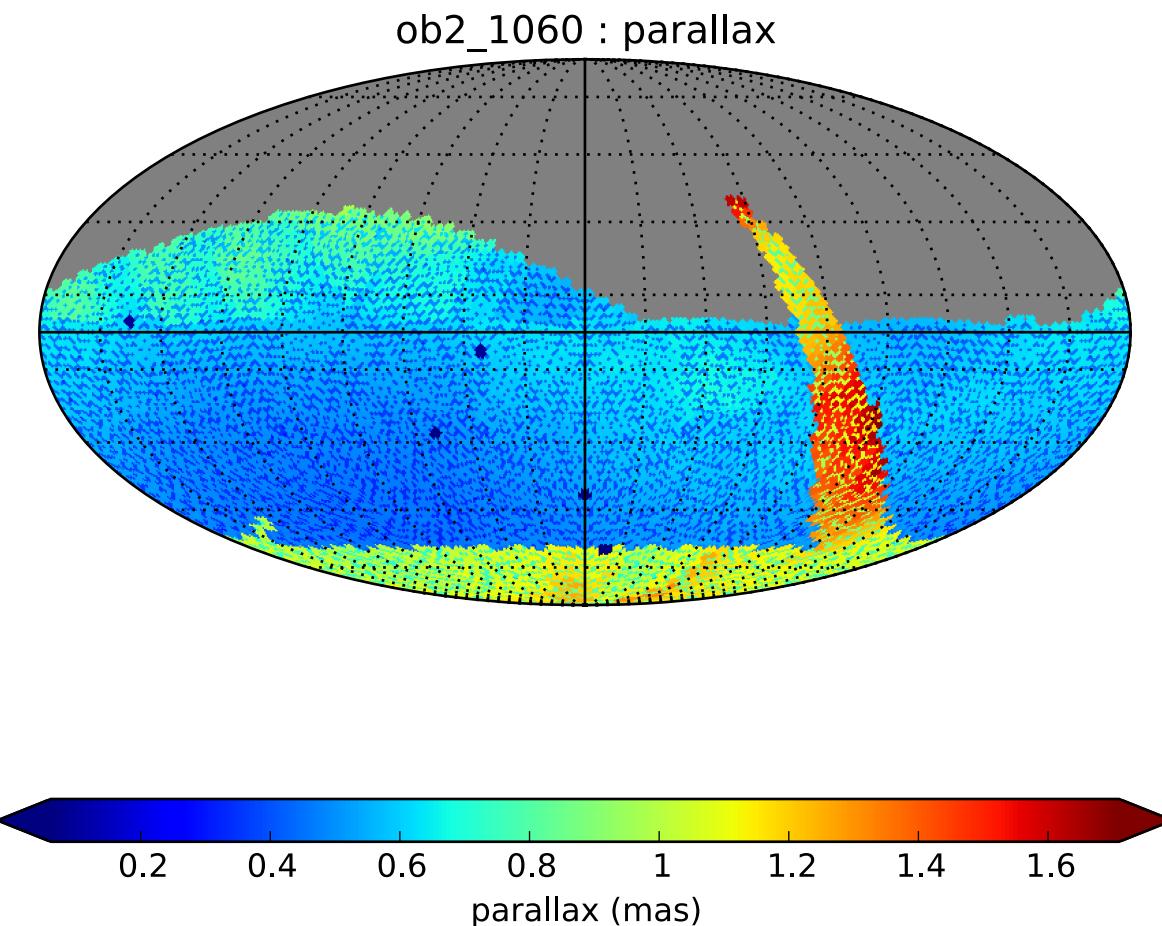
Slicer + metric summary



- UniSlicer + metric = single number (mean seeing)
- OneDSlicer + metric = histogram (or something per night)
- HealpixSlicer + metric = skyplot, histogram, power spectra



Code is designed to be extensible



*Add new stackers, slicers,
and metrics and mix and
match*

Astrometric precision for a
20th mag star

Add a stacker to compute
parallax factor for each visit

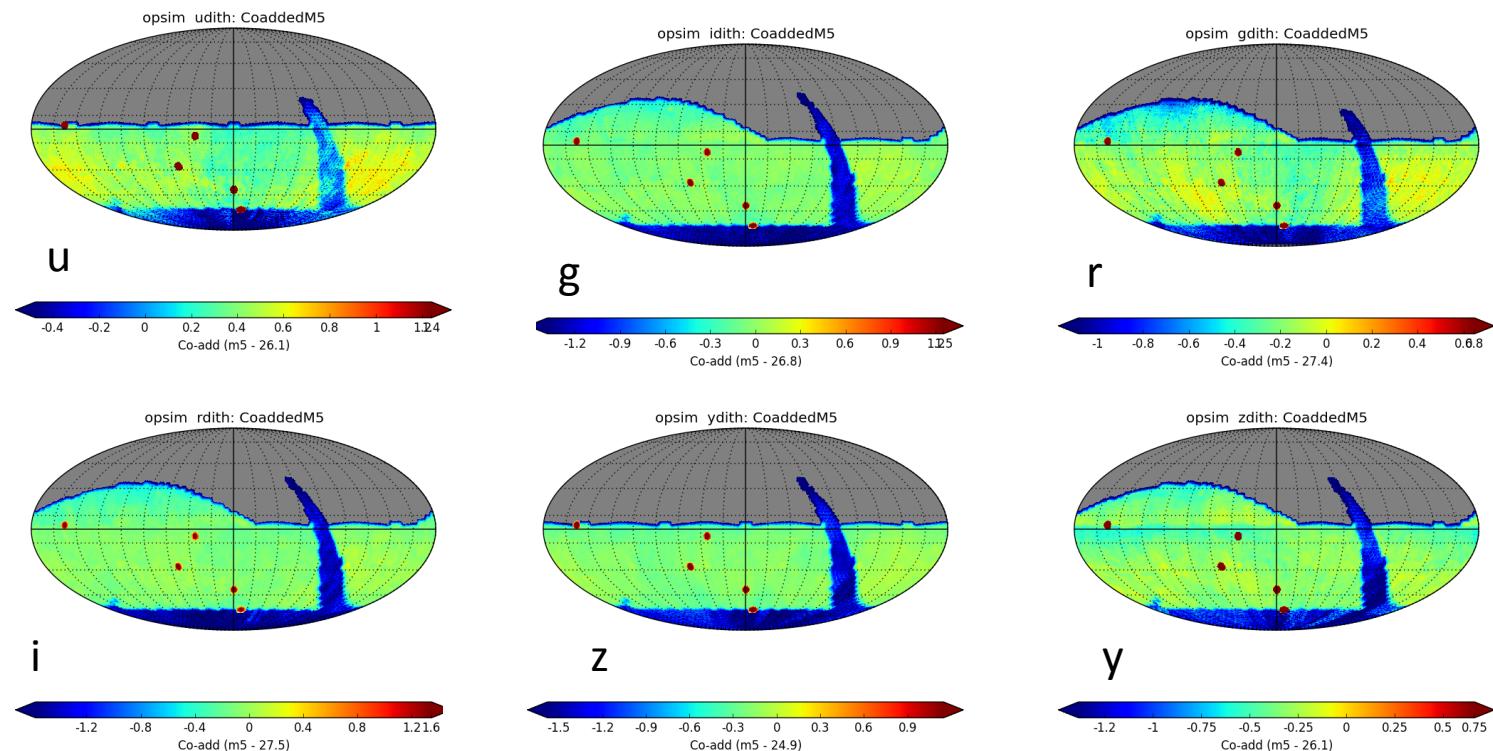
Then use seeing and sky
brightness to compute
astrometry errors in single
frames



Example MAF Output



- Can create large reports with MAF
 - Number of Visits, coadded depth, median single-visit depth, median airmass, median seeing
 - Per filter, Per proposal ID





One way to think about the Healpix grid

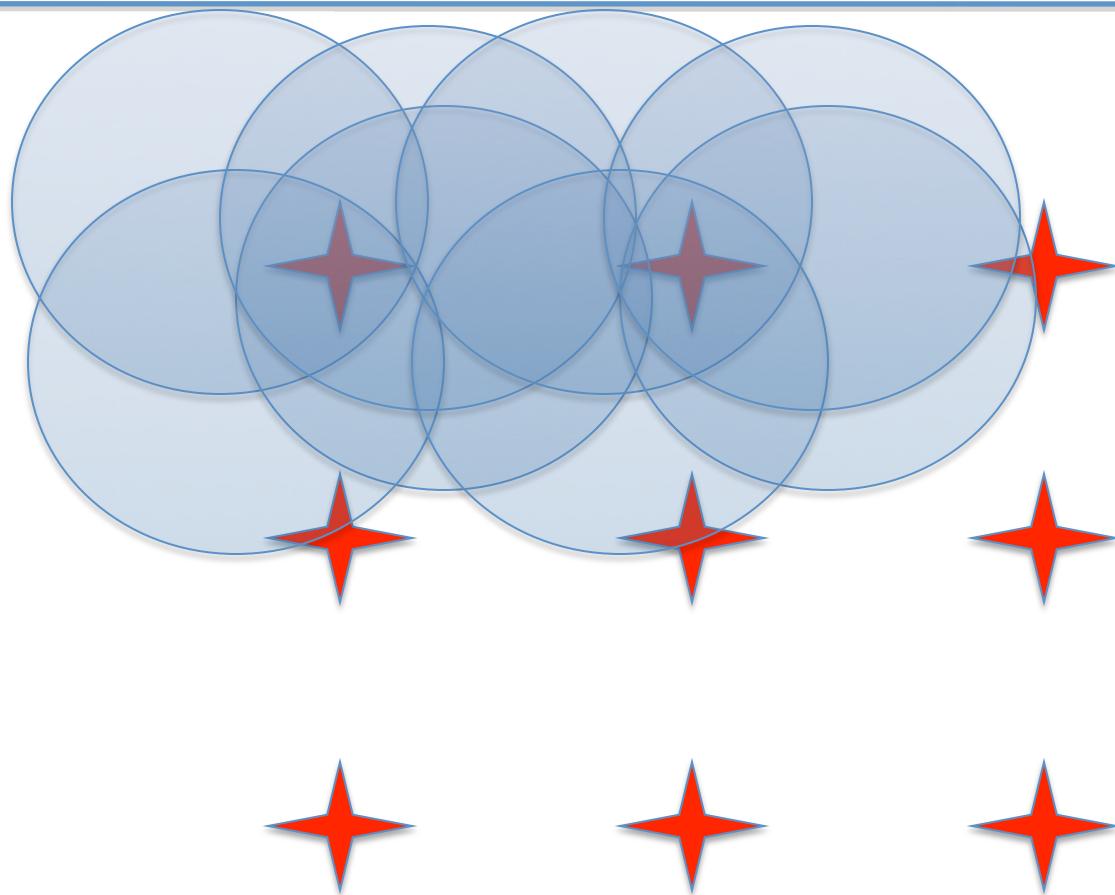


Which observations overlap star/galaxy/healpixel #1?

Pass those to the metric algorithm and save the result

Which observations overlap star/galaxy/healpixel #2?

Pass those to the metric algorithm and save the result.





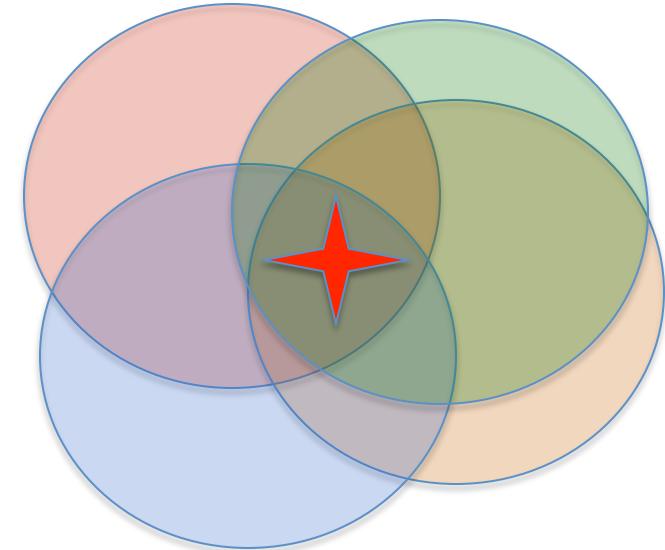
How well can we measure X given a sequence of observations?



Given the list of observations (dates, filters, seeing, etc.) of your favorite type of object, can we quantify how well we can do a science?

Given those observations:

- X% of z=1.5 supernova would be discovered
- The proper motion fit for a star would have a precision of X milliarcseconds
- The shape of a galaxy could be recovered to X%
- We could get follow-up spectra on X of the observations



We need the algorithms to convert an observation list into a science result.



Metrics we have developed already



- Proper Motion and Parallax fit uncertainties
- Transient light curve detection and characterization
- Period fitting
- NEO detection distance

All of these can be sub-classed and extended to write your own metrics, you don't have to start from scratch!



- **OpSim:** The Operations Simulator that outputs a database of simulated observations
- **Slicer:** The part of MAF that slices data correctly to make histograms, sky maps, etc.
- **Metric:** The algorithm that is computed on each slice from the slicer
- **Stacker:** Adds a pseudo-column to the OpSim
- **Healpixel:** A single pixel on an equal area tessellation of the sky.
- **Summary Statistic:** A metric that gets computed on the results of all the slice points (e.g., a mean-of-means across the sky).



Conclusions



- The essence of MAF
 - Select a set of OpSim observations using a SQL query
 - Use a slicer + metric to analyze the resulting set of observations



Example iPython Notebooks



Github repo: <http://ls.st/nm3>

Introductory Notebooks

IntroductionNotebook.v3.ipynb: Our introductory notebook, including installation instructions.

Slicers.v3.ipynb: Shows how the same metric can be run with three different slicers to produce different outputs.

TestNotebook.v3.ipynb: Test that ipython is working and you can load the MAF modules.

Advanced Capabilities

Stackers.v3.ipynb: Example of calculating and adding a new column to the OpSim output "on-the-fly" with MAF. Here, we calculate the hour angle of each pointing and then look at the distribution.

WritingNewMetric.v3.ipynb: A guide on how you can write and add your own algorithm into MAF by sub-classing an existing metric.

UsingMetricBundleData.v3.ipynb: Example of saving/restoring analysis and then adjusting and combining plots.

Static Science Notebooks

DepthGoodSeeing.v3.ipynb: Using SQL constraints to compare coadded depth and coadded depth of only visits taken under good seeing conditions.

DepthPerYear.v3.ipynb: An example of using SQL constraints to see the evolution of the coadded depth through the survey.

Dithers.v3.ipynb: MAF includes several potential dithering schemes. This notebook demonstrates those dither patterns and compares their effect on the resulting power spectrum.

NVisitsCoadd_AllFilters_DitherComparison.v3.ipynb: Examines the number of visits in each filter as well as comparing dithered and undithered surveys.

Solar System and Astrometry Notebooks

AstrometryMetrics.v3.ipynb: Example of running our proper motion and parallax metrics.

Transient Notebooks

TransientMetric.v3.ipynb: Example of our transient metric where you can set a simple transient light-curve shape and set a variety of detection criteria to measure the fraction of such transients LSST can recover.

Variable Notebooks

Our variable metrics are in the sims_maf_contrib repo. A notebook example of them can be found here.