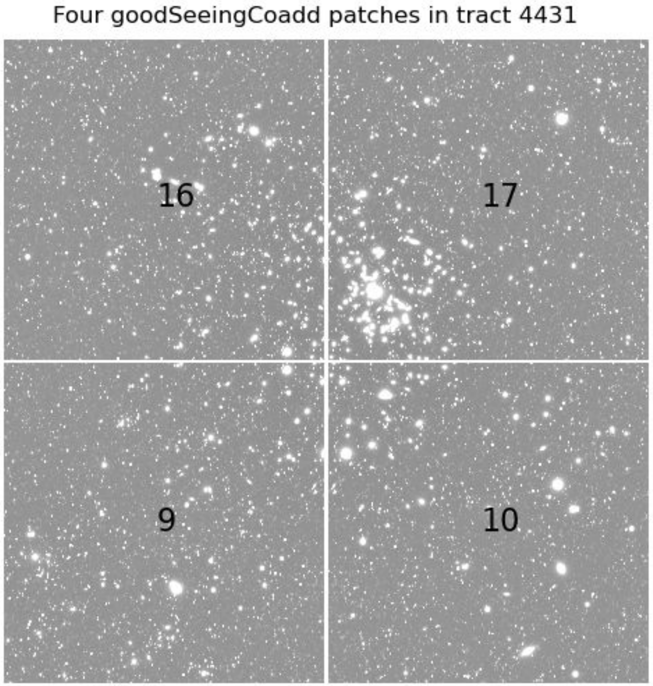
VERA C. RUBIN OBSERVATORY

May 2022 AP Performance Sprint

Eric Bellm, John Parejko, Meredith Rawls

We tested ApPipe executing time on DC2 images



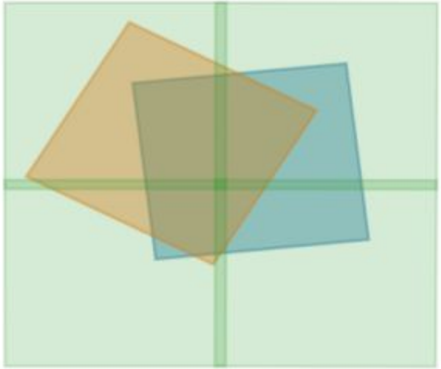
- Selected four `goodSeeingCoadd` patches in tract 4431 (lots of visits!) for this investigation
- DC2 doesn't have much variation of source density, but we picked patches containing a slightly dense galaxy cluster region anyway
- Templates on `lsst-devl` in `/repo/dc2`
  - Originated in: `u/kherner/2.2i/runs/tract4431-w40`
  - Curated to: `u/mrawls/DM-34827/coadd/4patch_4431`

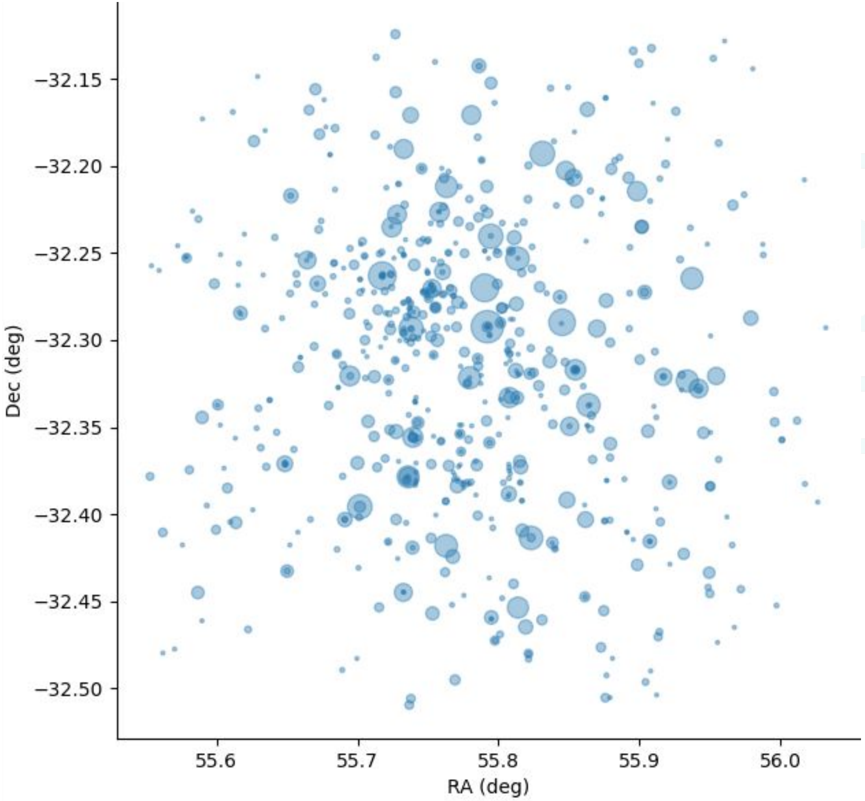
https://confluence.lsstcorp.org/display/DM/May+2022+Performance+Sprint+Summary

Dataset is 272 visits fully overlapping 4 patches in all bands

- Wrote a script to identify visit+detector datasets that fully fall inside this region
- Guarantees full template coverage
- Also yields more overlaps near center

```
/repo/dc2/u/mrawls/DM-34827/defaults/4patch_4431
```

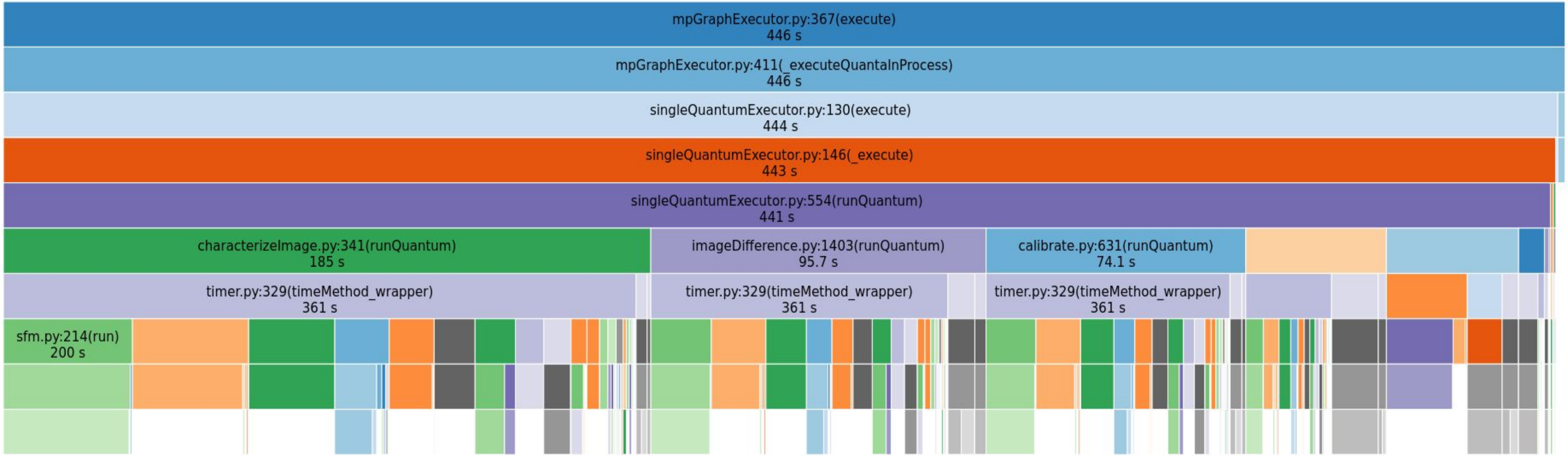




Before any changes, ApPipe took 446 s

Snakeviz profile (on a single visit+detector dataset) at the start of our sprint

`CharacterizeImage` was the obvious place to start optimizing



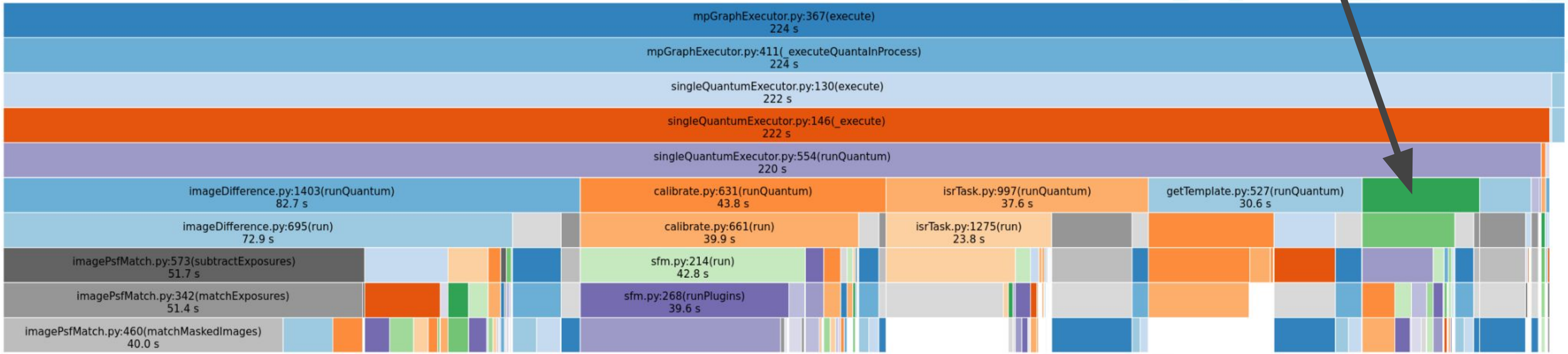
After our sprint, ApPipe took 224 s on the same data

Snakeviz profile (on a single visit+detector dataset) after removing unnecessary plugins and using `psfex` instead of `piff`

Loading files takes ~30 s of this, which should not matter for prompt processing with preload and an in-memory Butler

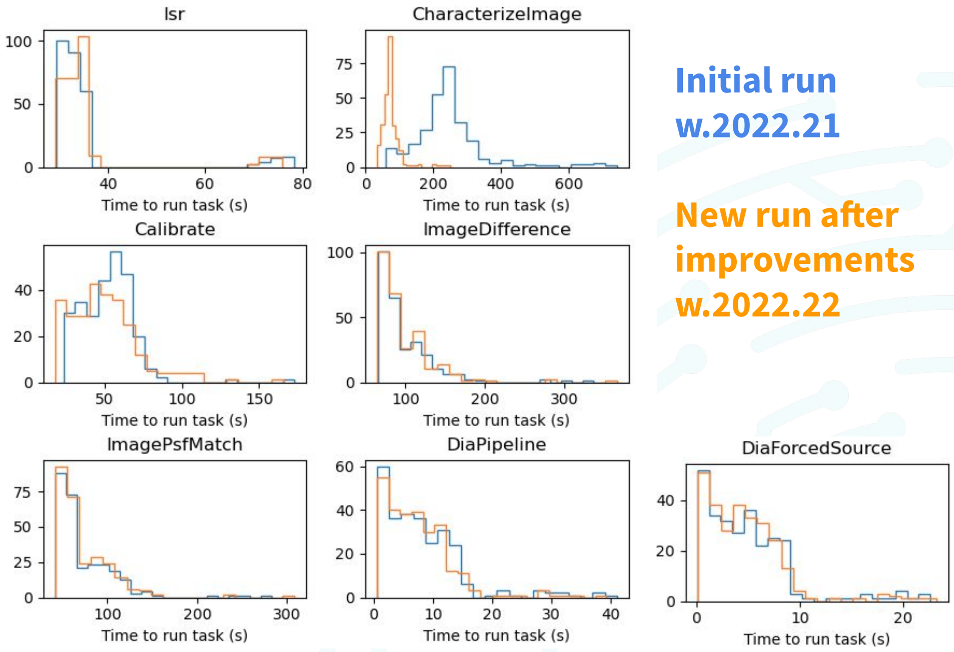
~200 s total runtime (with caveats!)

`CharacterizeImage`



Biggest runtime improvement was in `CharacterizeImage`

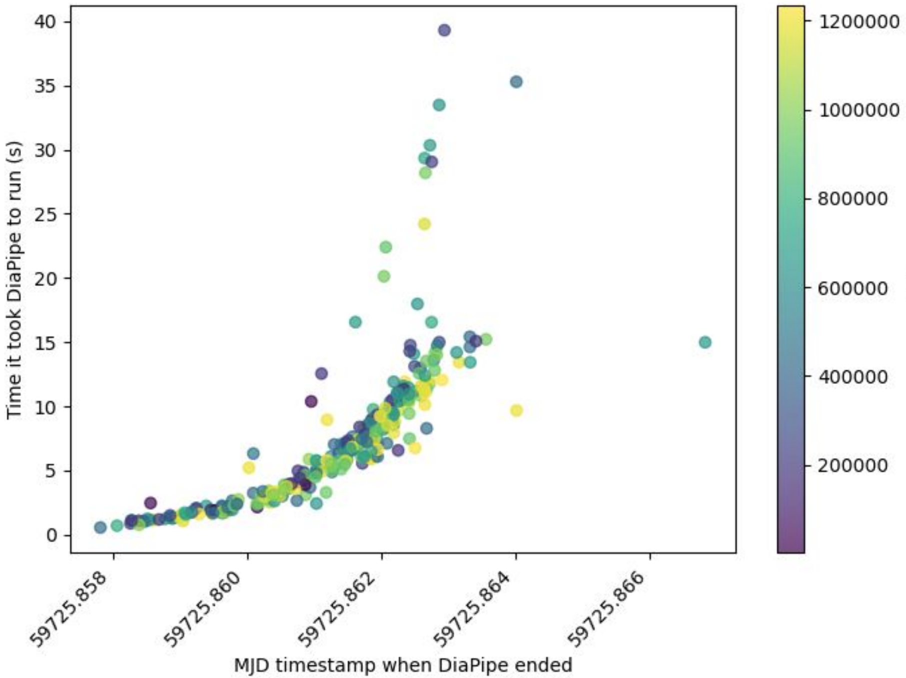
- bps runs on `lsst-devl01`
- ApPipe is three main steps
  - ○ Single frame measurement (`ISR, Characterize, Calibrate`)
  - ○ Template convolution and subtraction (`ImagePsfMatch, ImageDifference`)
  - ○ Associate sources & do forced photometry (`DiaPipe, DiaForcedSource`)
- Naively, our longest step should be `ImageDifference`, where we perform the most convolutions
- Need to explore timing outliers (e.g., bimodal ISR)

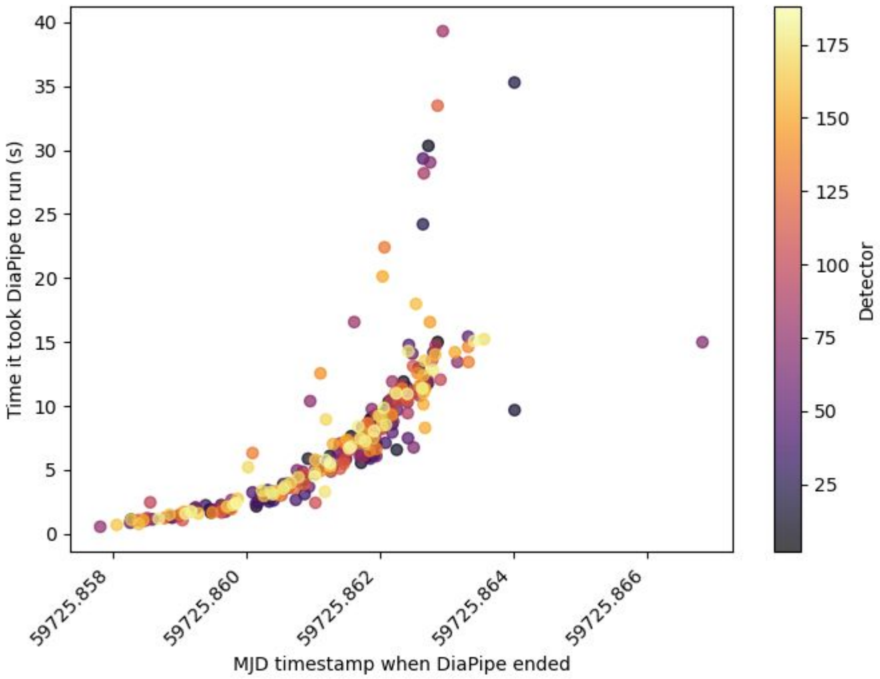


Initial run w.2022.21

New run after improvements w.2022.22

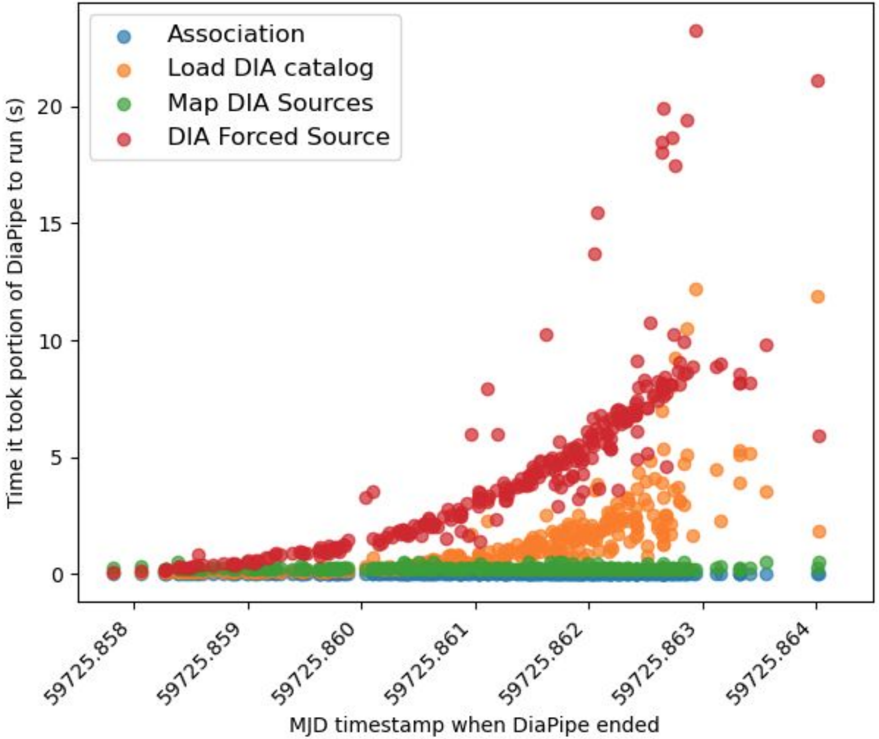
DiaPipe slows over time, with no dependence on visit/detector





Slowest parts of DiaPipe are forced photometry (and loading the DIA catalog)

- The earlier snakeviz profiles don't measure DiaPipe — they were run for a single visit+detector dataset
- Don't yet know whether "DIA Forced Source" scales as O(N^2)
- Not yet sure if database loads or cross-matching dominate runtime



Sprint accomplishments

- Identified and removed `CharacterizeImage` plugins that are unnecessary for ApPipe, resulting in a ~30% improvement in runtime
- Identified `PiffPsfDeterminer` as a substantial contributor to `CharacterizeImageTask` time; plan to switch back to psfex, which should be sufficient for our needs
- Developed improved datasets and tooling for future performance optimization
- First look at `DiaPipeTask` timing performance at scale
- John P. guesses that we could gain another ~30s with “easy” cleanups/disabling other unnecessary measurements, before we need to take a hard look at algorithms

Future work

- Daily performance monitoring on a new DC2 CI “ap_verify-style dataset” (two r-band visit+detector datasets from the larger run, WIP)
- Test with updated image differencing code (still being integrated with ap_pipe)
- Line-by-line profiling to drill down into the slowest parts of each task
- Efficiency improvements to `CharacterizeImageTask` (RFC-857)

- Testing in the operational environment
  - USDF hardware
  - Production APDB with 12 months of DIA Source history
  - Prompt Processing with preload