# SCHEDULING ZTF

Eric Bellm

March 19, 2018

## What is the scheduling problem we're trying to solve?

Decide what fields to observe in what order.

*while maintaining the desired cadences*
*while maximizing image quality*
*while avoiding the moon*
*while minimizing slew time*
*while minimizing filter changes*
*while interleaving multiple observing programs*
*while equalizing observing time between programs*
*while responding to TOOs*
*while recovering from weather losses*

45 seconds per exposure over a 10-hour night:
800 exposures/night ⇒ **800! ≈ $10^{1977}$ possible orderings**

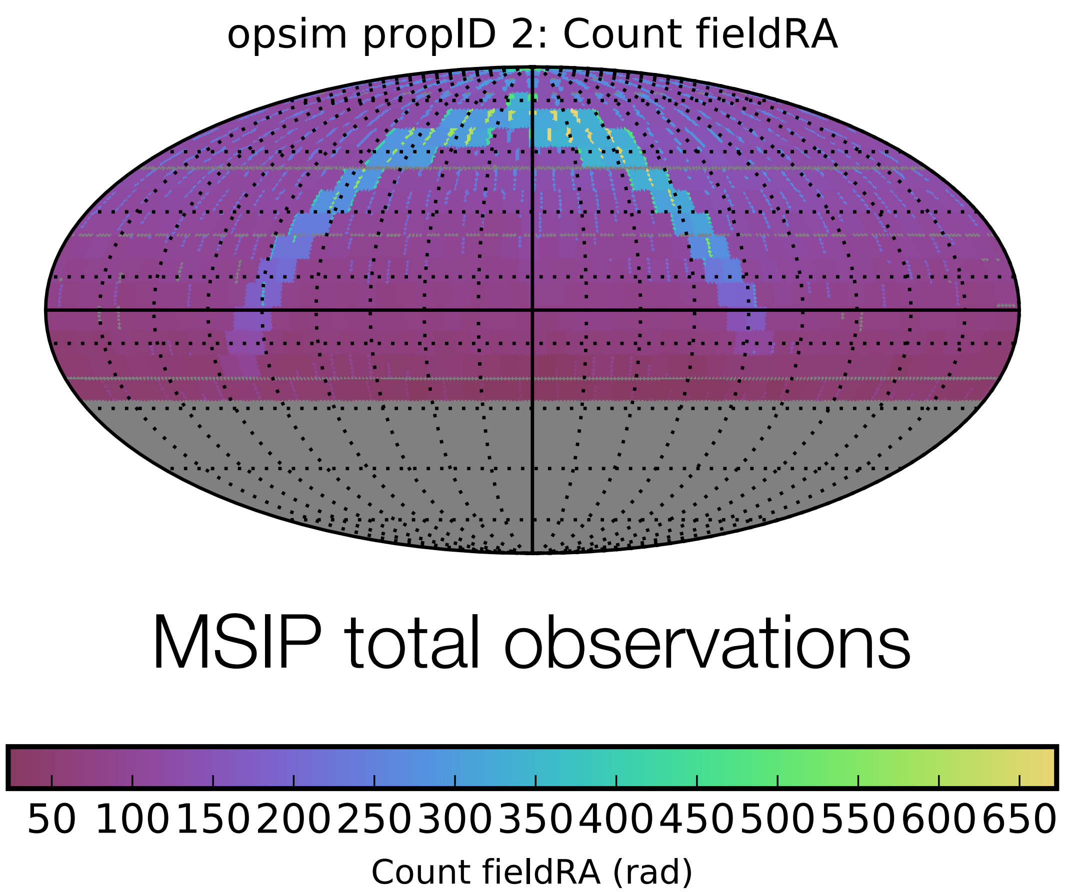
NP hard!

# ZTF is conducting an unusually large number of surveys simultaneously.

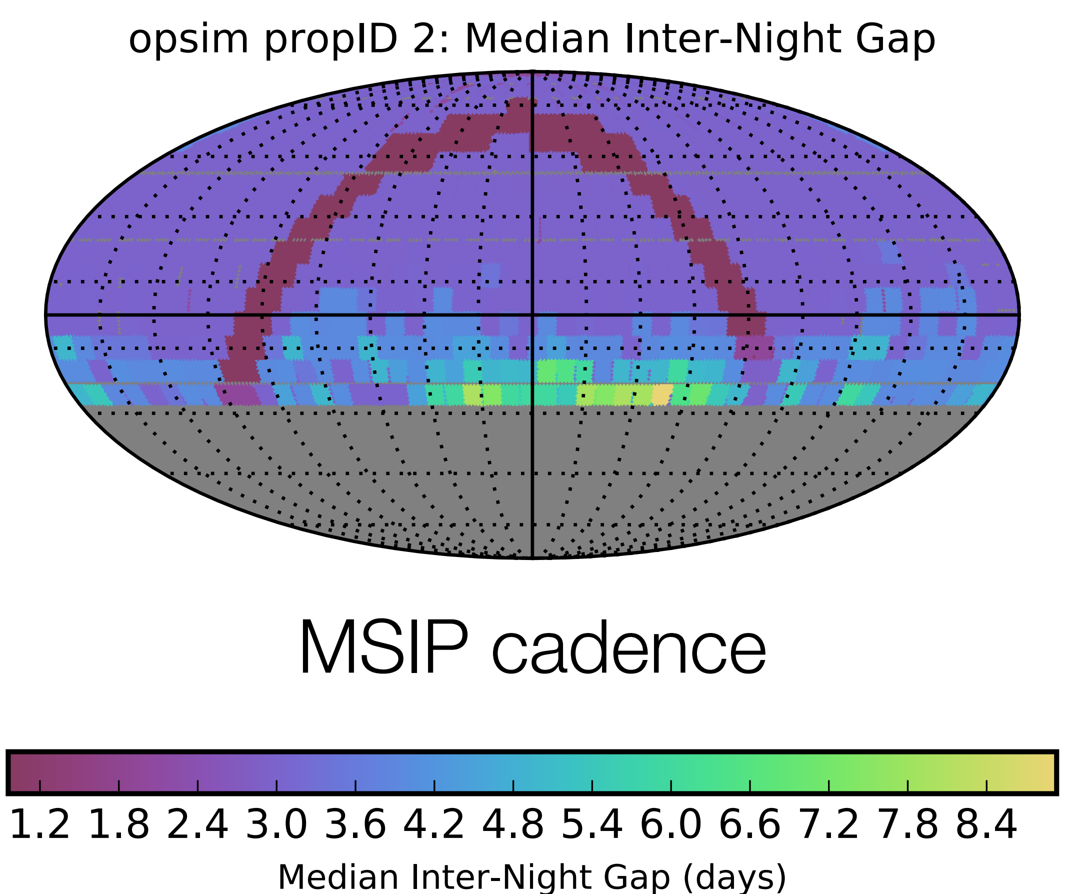
## MSIP
Northern Sky Survey
Galactic Plane Survey

## Partnership
High Cadence
i-band
HiCAPS
Twilight Survey
SFR survey
TOOs

## Caltech
multiple rotating programs…




MSIP total observations




MSIP cadence

# The simplest scheduler is a static list.

```
[(time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time),
 (time, field, filter, exposure time)]
```

**Pros**

conceptually simple
technically robust

**Cons**

labor intensive
hard to reproduce

# Greedy schedulers always take the next best option.

Recompute score before each new observation and observe the field with the highest value.

[ (field, score),
(field, score),
**(field, score),**
(field, score),
(field, score),
(field, score),
(field, score),
(field, score),
(field, score)]

# Greedy schedulers always take the next best option.

Recompute score before each new observation and observe the field with the highest value.

[ (field, score),
(field, score),
**(field, score),**
(field, score),
(field, score),
(field, score),
(field, score),
(field, score),
(field, score)]

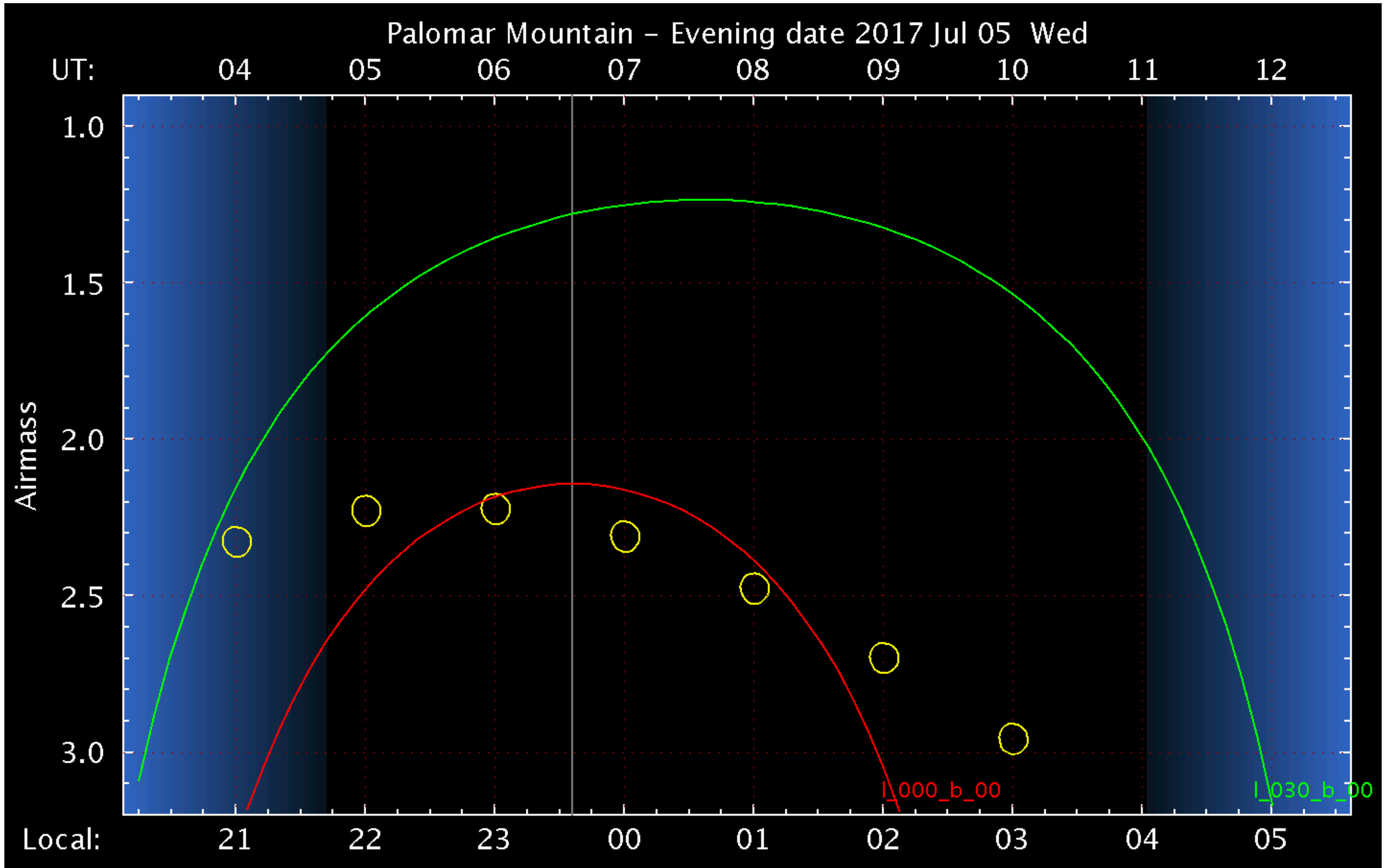
**Pros**

simple to implement

provides some optimization

**Cons**

no lookahead; cannot optimize globally



# We can optimize ZTF for discovery.

Maximize volume surveyed per image:

$$V = \frac{4\pi}{3} d^3$$

$$\propto 10^{0.6 m_{\rm lim}}$$

(to maximize SNR, use $10^{0.8 m_{\rm lim}}$)

Limiting magnitude depends on:

filter, sky brightness, airmass, seeing

So: **maximize** **the *volume-weighted* number of images observed** **in acceptable cadence windows.**

# We can optimize ZTF for discovery.

Maximize volume surveyed per image:

$$V = \frac{4\pi}{3} d^3$$

$$\propto 10^{0.6 m_{\rm lim}}$$

(to maximize SNR, use $10^{0.8 m_{\rm lim}}$)

Limiting magnitude depends on:

filter, sky brightness, airmass, seeing

optimization algorithm

objective function

So: maximize the ***volume-weighted*** number of images observed in acceptable cadence windows.

observing strategy

# We can optimize ZTF for discovery.

Maximize volume surveyed per image:

$$V = \frac{4\pi}{3} d^3$$

$$\propto 10^{0.6 m_{\rm lim}}$$

(to maximize SNR, use $10^{0.8 m_{\rm lim}}$)

Limiting magnitude depends on:

filter, sky brightness, airmass, seeing

optimization algorithm

objective function

So: maximize the ***volume-weighted*** number of images observed in acceptable cadence windows.

observing strategy

For greedy schedulers, choose highest $\dot{V} \propto \frac{10^{0.6 m_{\rm lim}}}{t_{\rm exp} + t_{\rm OH}}$

# We compute volume-weighting with a predictive model.

compute sky brightness (-> limiting mag)

xgboost model trained on PTF using the following features:

- filter (g, r, or i)
- sun altitude
- altitude, azimuth
- moon illumination fraction
- moon altitude
- moon distance

# Acceptable cadence windows are defined per observing program.

Each observing (sub-)program specifies

- a survey footprint
  (either as discrete field ids or a selection function)
- a minimum number of days between revisits
- a "request set" per field: the number of observations per filter to be observed tonight
- the fraction of total time to be devoted to this subprogram

Intra-night separations are set to be simply larger than a fixed interval (30 minutes) to enable asteroid discrimination

(previous efforts to define on-off cadence windows based on time since last observations proved too brittle.)

Each night, each program proposes request sets according to their observability and inter-night cadence, and the scheduler constructs an optimal observing plan from among them.

# The MSIP observing programs provide a simple example.

```
{"program_name": "MSIP",
 "subprogram_name": "all_sky",
 "program_observing_fraction": 0.4,
 "subprogram_fraction": 0.85,
 "field_selections":{"dec_range":[-31,80],
                     "abs_b_range":[7,90],
                     "grid_id":0},
 "filter_ids": [1, 2],
 "internight_gap_days": 3,
 "n_visits_per_night": 2},
{"program_name": "MSIP",
 "subprogram_name": "nightly_plane",
 "program_observing_fraction": 0.4,
 "subprogram_fraction": 0.15,
 "field_selections":{"dec_range":[-31,80],
                     "abs_b_range":[0,7],
                     "grid_id":0},
 "filter_ids": [1, 2],
  "internight_gap_days": 1,
  “n_visits_per_night": 2}
```

# A grid approach enables a nightly solution.

**Time Blocks**

**Request Sets (Fields)**

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| $r_0$ (gggg) | | | | | | | | | |
| $r_1$ (gggg) | | | | | | | | | |
| $r_2$ (gr) | | | | | | | | | |
| $r_3$ (gr) | | | | | | | | | |
| $r_4$ (i) | | | | | | | | | |
| $r_5$ (grg) | | | | | | | | | |
| $r_6$ (rgr) | | | | | | | | | |

# We use Integer Programming techniques to perform nightly optimization.

$V_{rtf}$ Volume factor for request field r at time t in filter f

$Y_{rtf}$ ("yes") =1 if we observe r at t in f, 0 otherwise

maximize $$\sum_{r \in R} \sum_{t \in T} \sum_{f \in F} V_{rtf} Y_{rtf}$$

subject to

$$\sum_{t \in T} Y_{rtf} \leq n_{rf} \; \forall \; r \in R, f \in F$$

number of requests in this set

$$\sum_{r \in R} Y_{rtf} \leq n_{\max} \; \forall \; t \in T$$

number of observations in this slot

And enforce one filter per slot + program balance

# A grid approach enables a nightly solution.

**Time Blocks**

**Request Sets (Fields)**

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| $r_0$ (gggg) | | | | | | | | | |
| $r_1$ (gggg) | | | | | | | | | |
| $r_2$ (gr) | | | | | | | | | |
| $r_3$ (gr) | | | | | | | | | |
| $r_4$ (i) | | | | | | | | | |
| $r_5$ (grg) | | | | | | | | | |
| $r_6$ (rgr) | | | | | | | | | |

# A grid approach enables a nightly solution.

**Time Blocks**

**Request Sets (Fields)**

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| $r_0$ (gggg) | | | | | | | | | |
| $r_1$ (gggg) | | | | | | | | | |
| $r_2$ (gr) | | | | | | | | | |
| $r_3$ (gr) | | | | | | | | | |
| $r_4$ (i) | | | | | | | | | |
| $r_5$ (grg) | | | | | | | | | |
| $r_6$ (rgr) | | | | | | | | | |

# A grid approach enables a nightly solution.

**Time Blocks**

**Request Sets (Fields)**

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| $r_0$ (gggg) | $Y_0=1$ | | | | | | | | |
| $r_1$ (gggg) | $Y_1=1$ | | | | | | | | |
| $r_2$ (gr) | $Y_2=1$ | | | | | | | | |
| $r_3$ (gr) | $Y_3=0$ | | | | | | | | |
| $r_4$ (i) | $Y_4=1$ | | | | | | | | |
| $r_5$ (grg) | $Y_5=1$ | | | | | | | | |
| $r_6$ (rgr) | $Y_6=0$ | | | | | | | | |

# A grid approach enables a nightly solution.

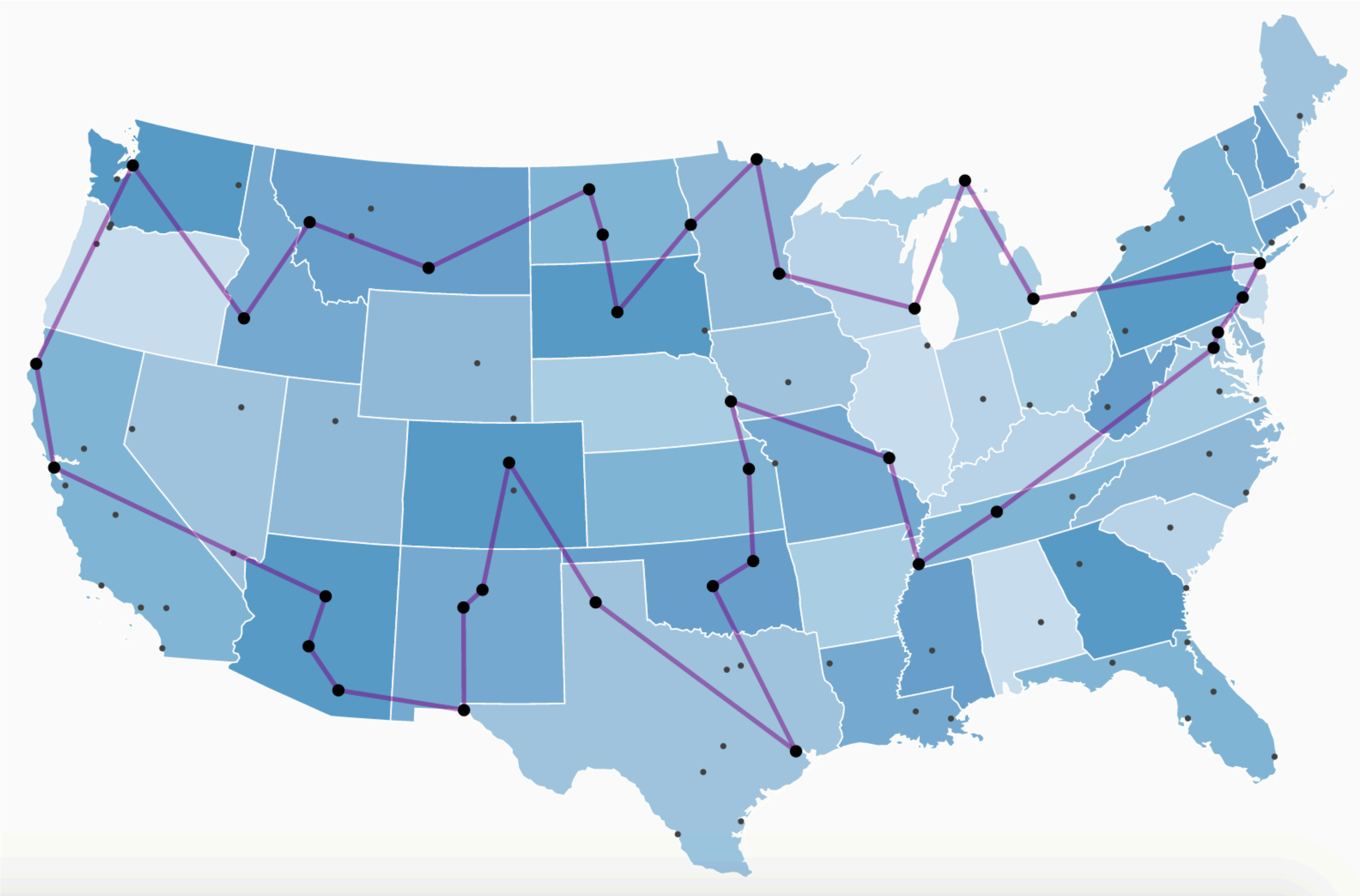
**Time Blocks**

**Request Sets (Fields)**

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|---|---|
| | | g | r | g | g | i | g | g | r |
| $r_0$ (gggg) | $Y_0=1$ | | | X | X | | X | X | |
| $r_1$ (gggg) | $Y_1=1$ | X | | X | X | | X | | |
| $r_2$ (gr) | $Y_2=1$ | | | | | | | X | |
| $r_3$ (gr) | $Y_3=0$ | | | | | | | | |
| $r_4$ (i) | $Y_4=1$ | | | | | X | | | |
| $r_5$ (grg) | $Y_5=1$ | | | | | | X | X | X |
| $r_6$ (rgr) | $Y_6=0$ | | | | | | | | |

## Then we sequence each block by solving the TSP.

Distances defined by slew time between requests in this block.



HA and Declination slews don't change with slot,
but dome slews do.

# Optimization requires tradeoffs!

**This scheduler is very good at:**

scheduling observations near zenith

maintaining cadence between nights

obtaining the requested number of observations

*(in fact, a field is only scheduled if its request set can be fully satisfied!)*

maintaining balance between programs

minimizing slew time*

**This scheduler cannot:**

guarantee filter order within the night

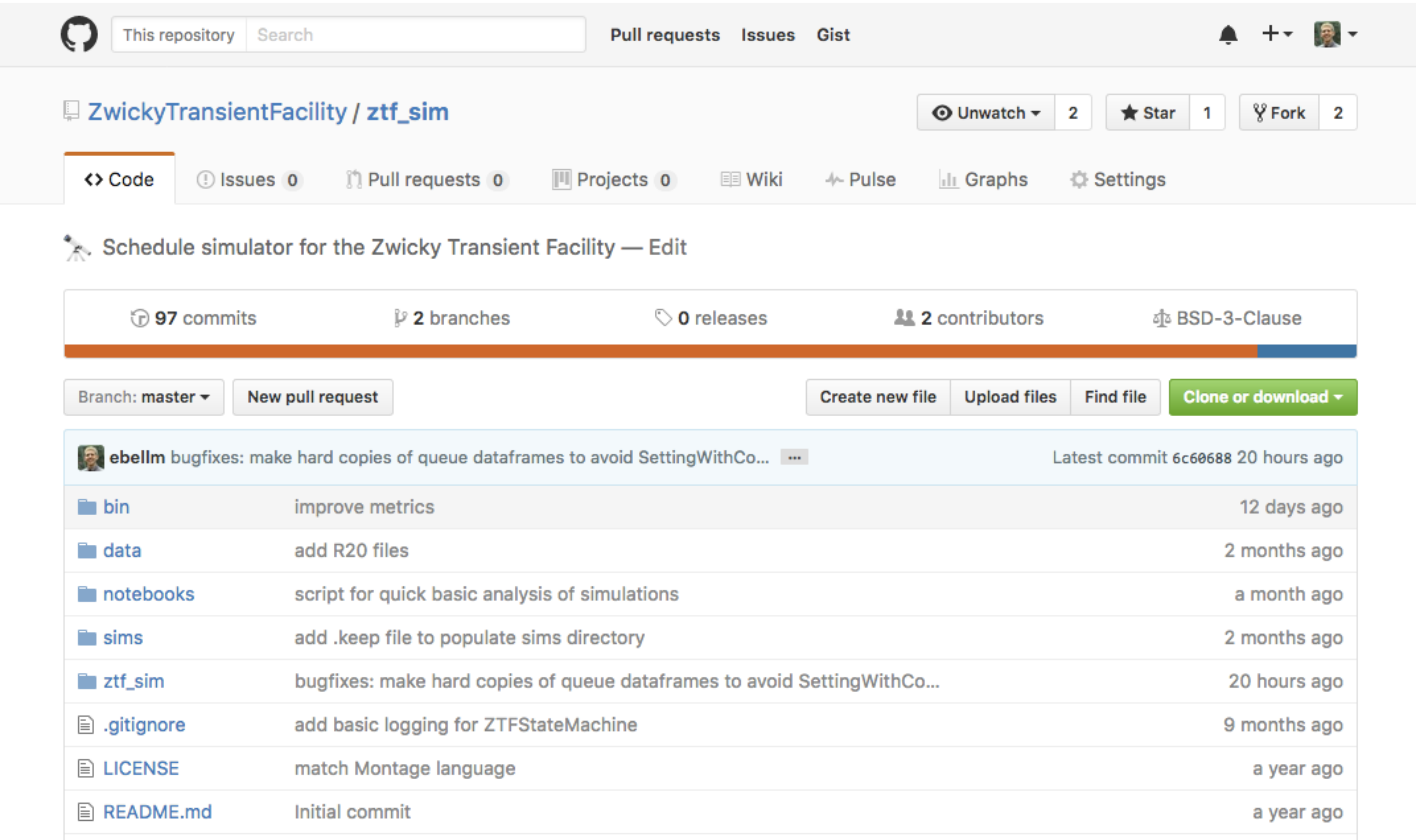
enforce exact times between observations

schedule continuous blocks of observations*

guarantee that there are observations at all times

# I’ve implemented this scheme in the production system.

https://github.com/ZwickyTransientFacility/ztf_sim



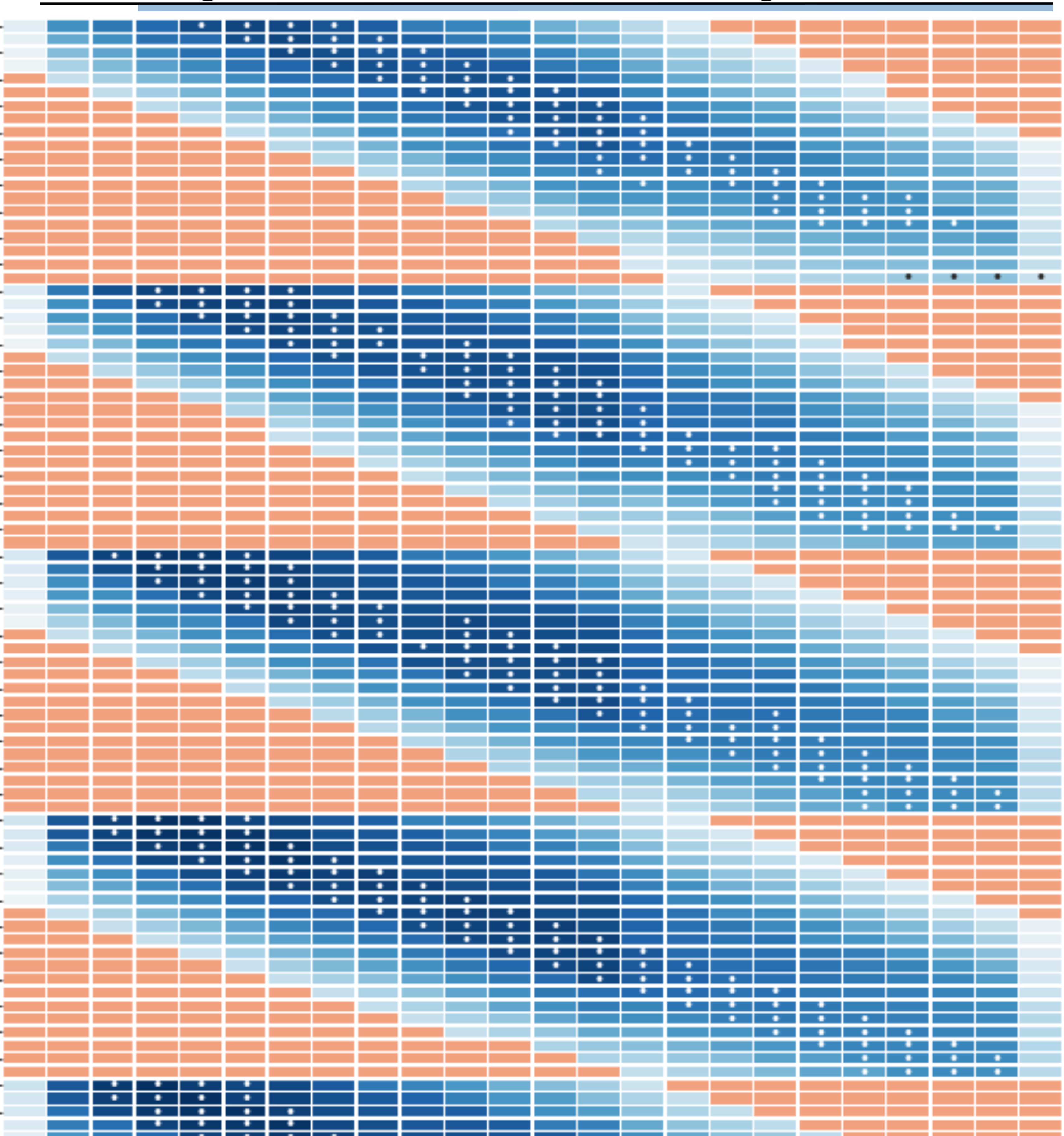
## The core functionality is complete.

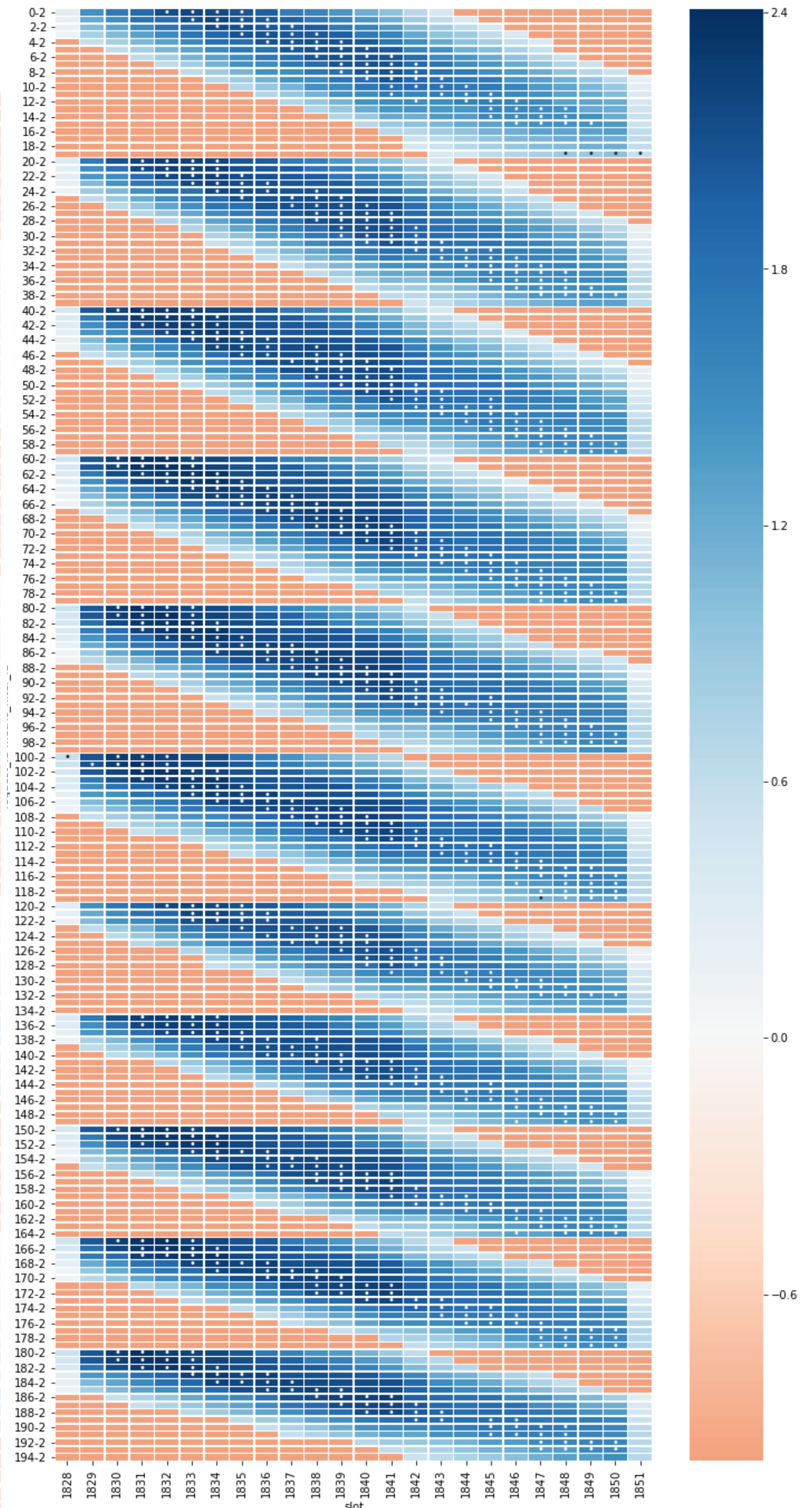
**swappable list, greedy, and slot-based schedulers**

runs in simulation mode
- weather losses (from PTF & iPTF)
- realistic telescope behavior
- exports pointing history to LSST MAF

integrated with the ZTF robotic system
- webserver client that uses the simulator as a library

# We’ve been running the scheduler throughout commissioning.

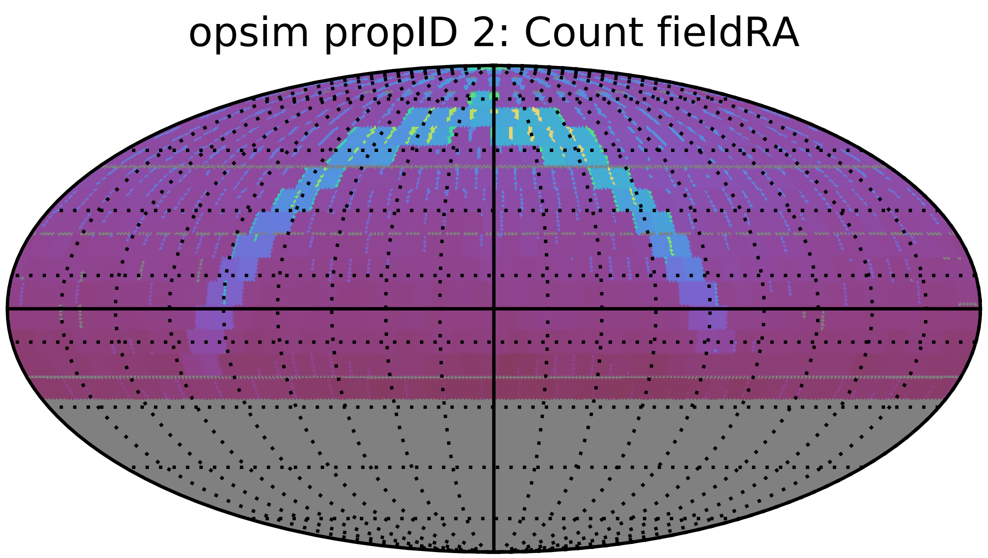




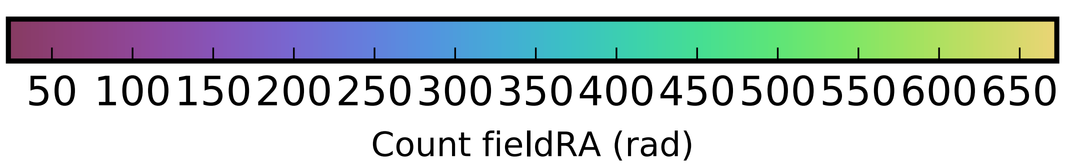
# A nine-month simulation shows good results.

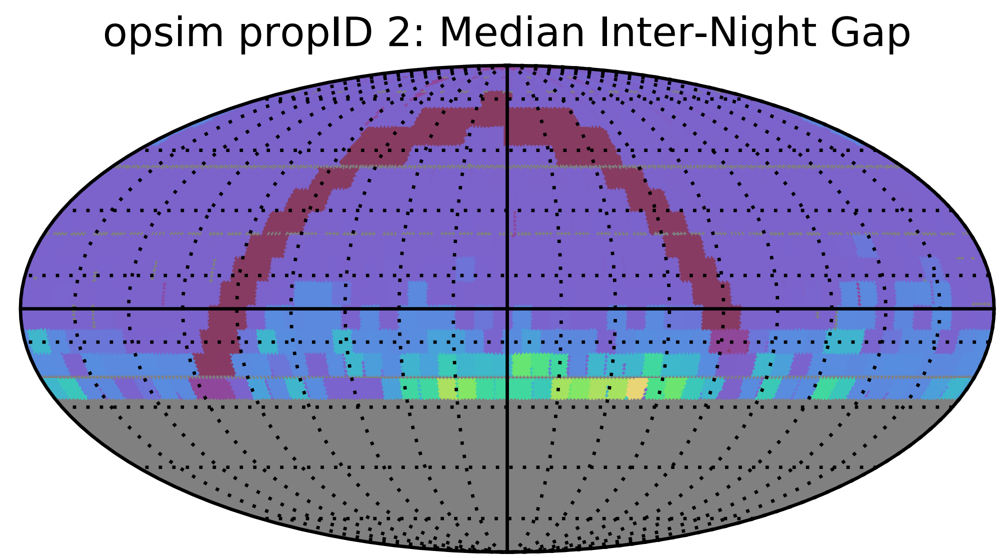
| | Fields | Intra-night cadence | Nightly cadence |
|---|---|---|---|
| **MSIP Northern Sky Survey** | All fields > Dec -31 & \|b\| > 7 | 3 days | 2 visits (gr) |
| **MSIP Galactic Plane** | All fields > Dec -31 & \|b\| < 7 | 1 day | 2 visits (gr) |
| **Collaboration High-Cadence** | 72 fields | 1 day | 6 visits (gggrrr) |
| **Collaboration i-band** | 249 fields | 4 days | 1 visit (i) |
| **Caltech TAC** | *All fields > Dec -31* | *1 day* | *2 visits (rr)* |

# Sky coverage is reasonable.

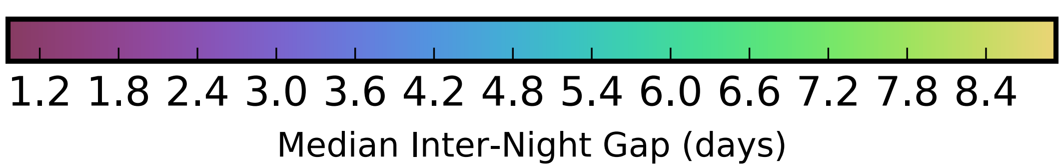


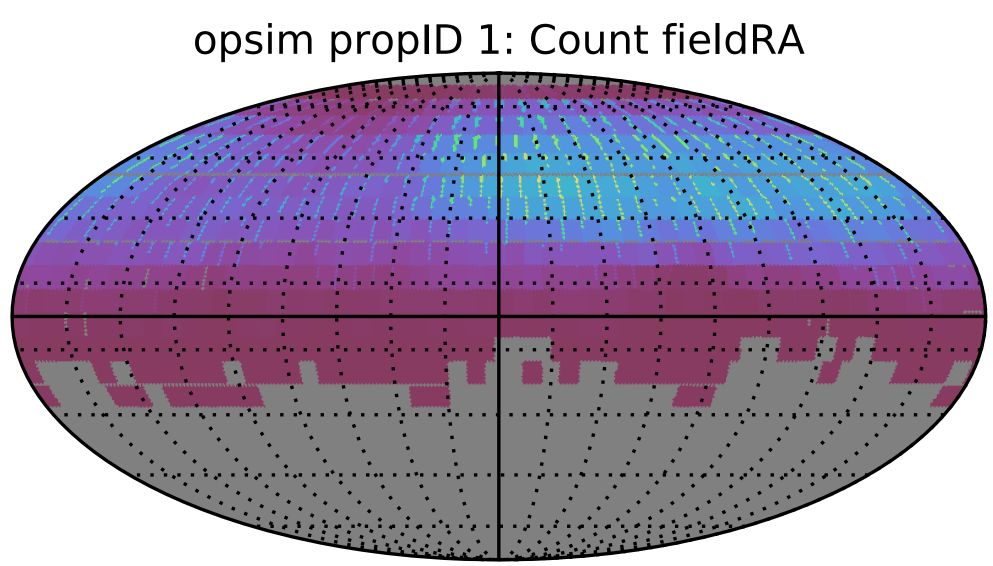

MSIP total observations



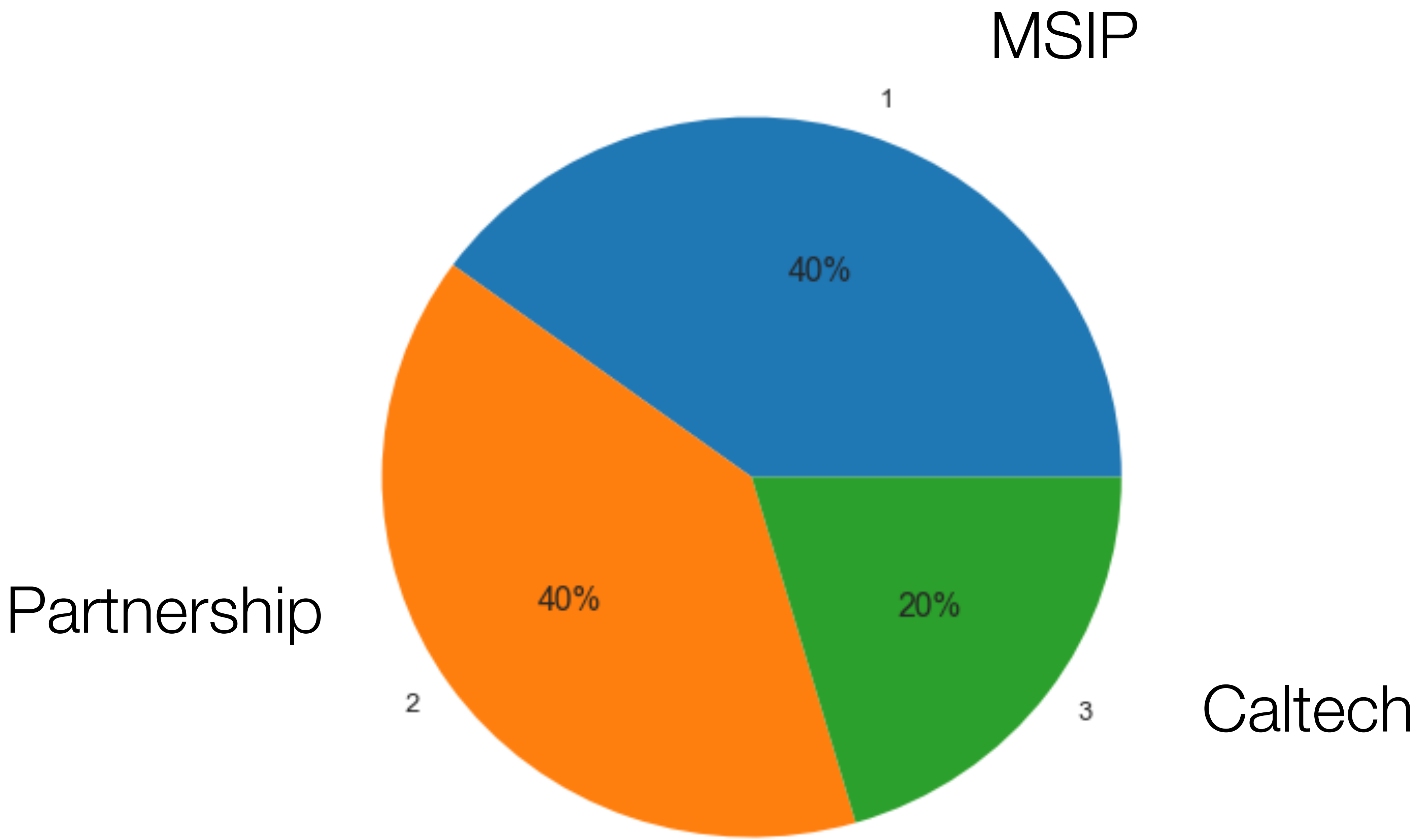

MSIP cadence



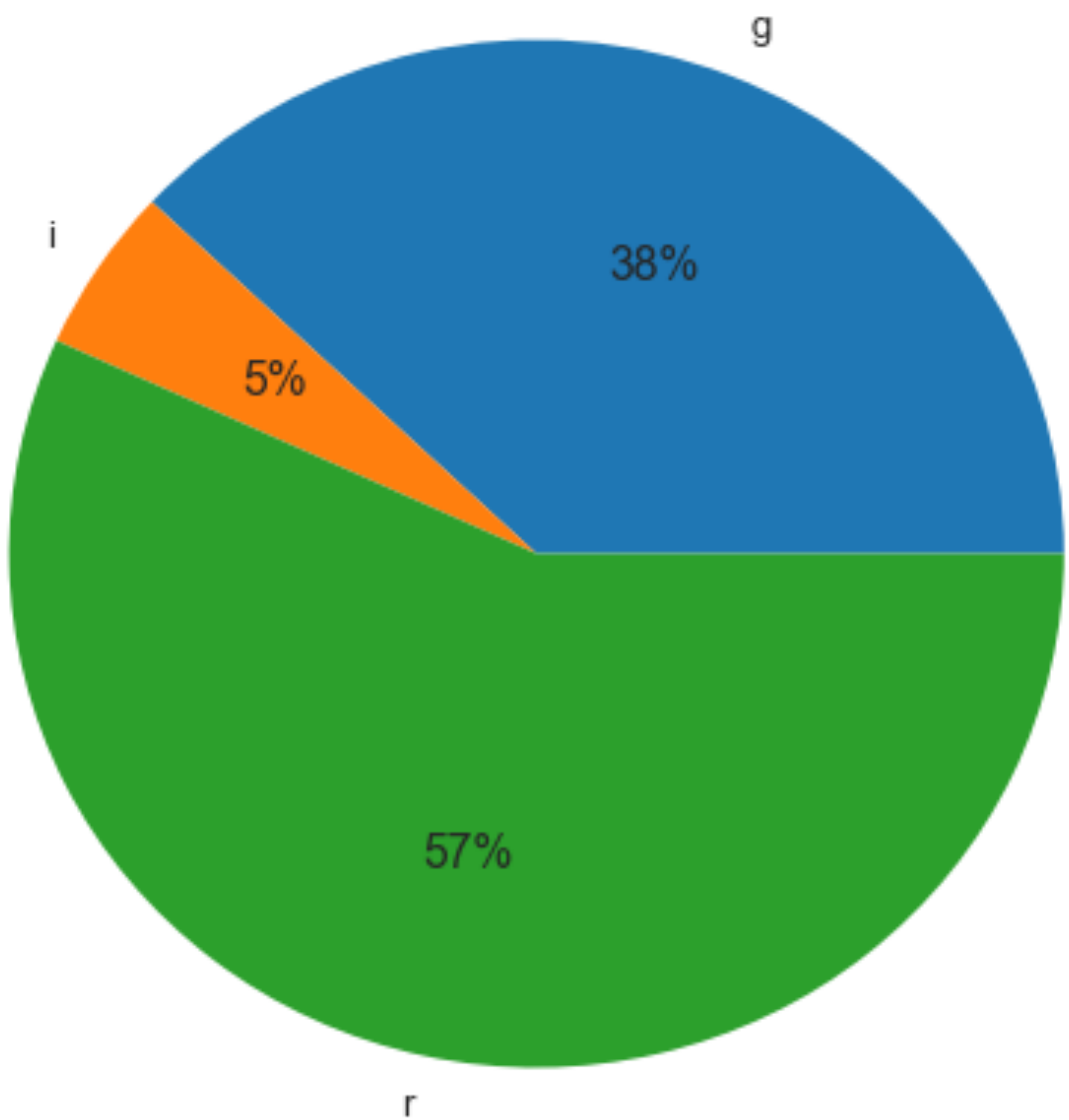




Collaboration total observations

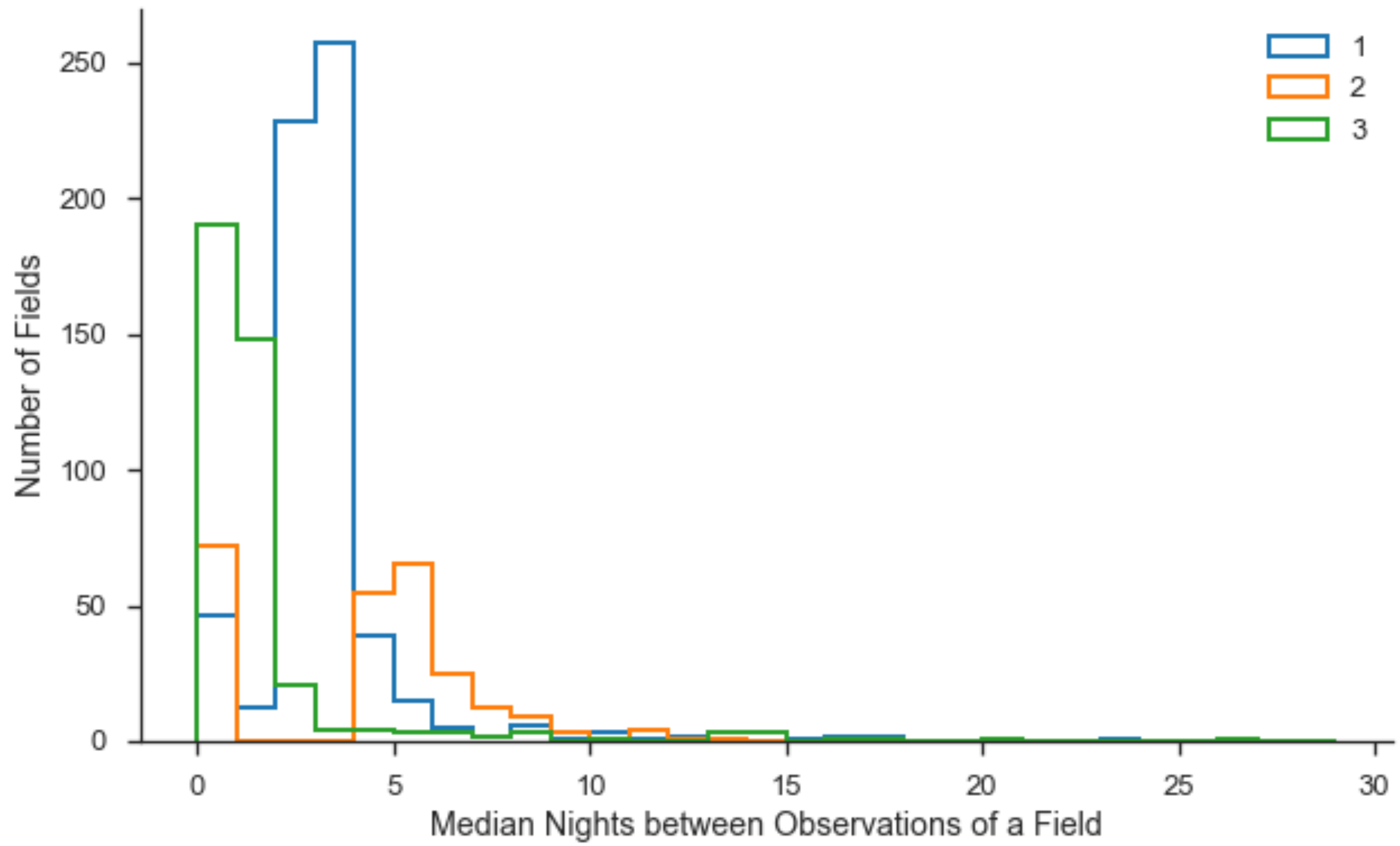
# Program balance is enforced correctly.

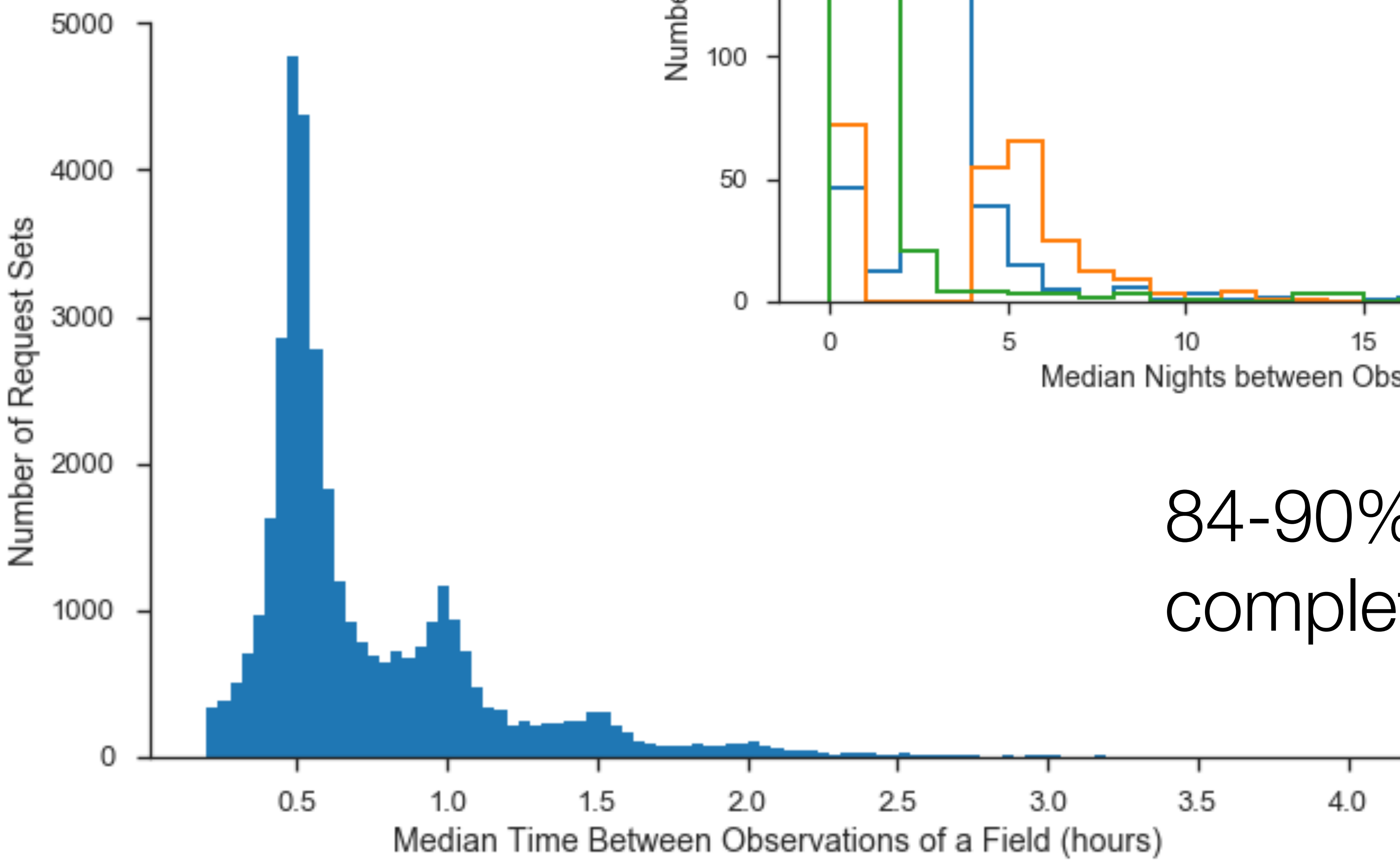


# i-band filter scheduling needs tweaking.



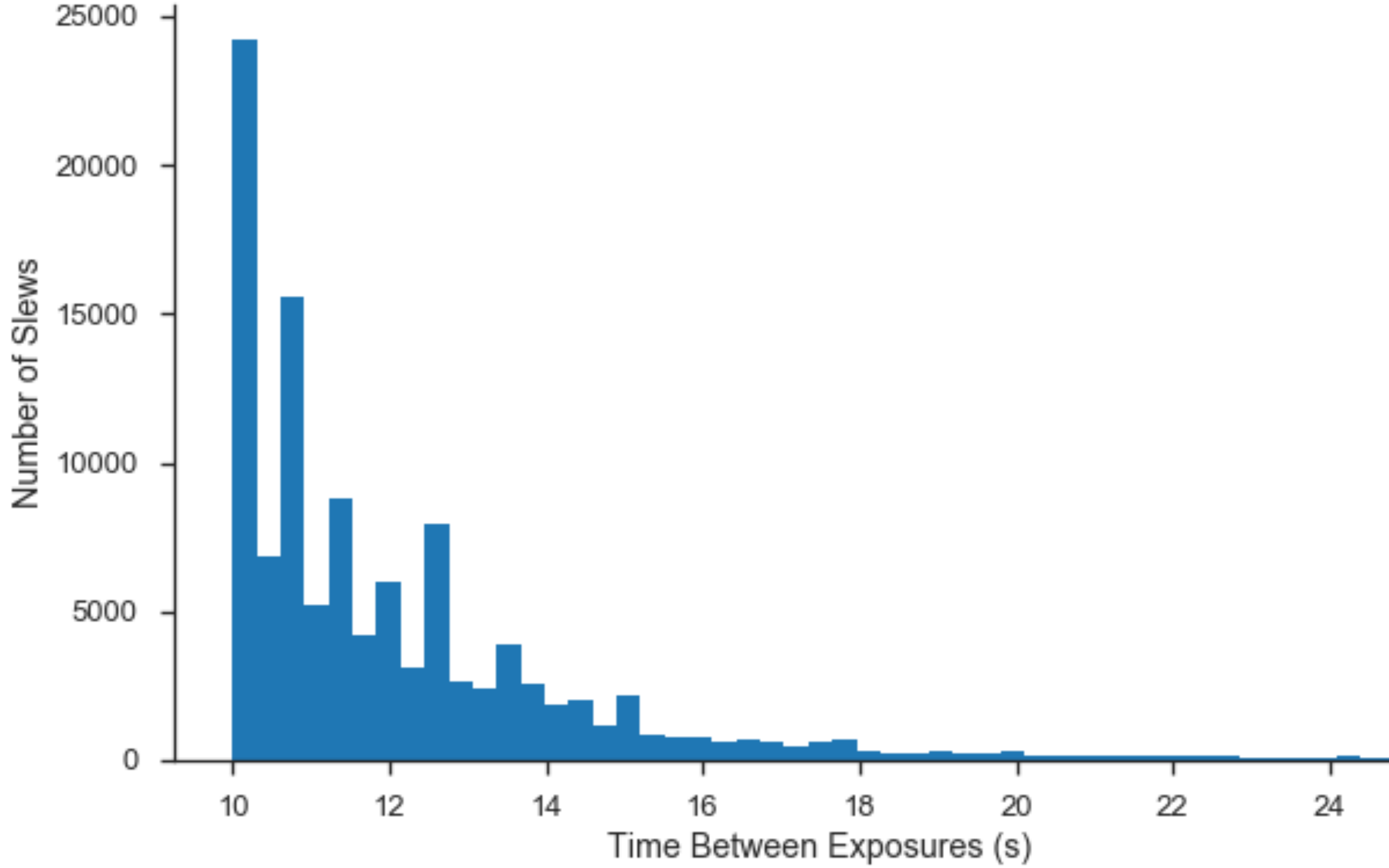
# Cadences requested are being scheduled correctly.

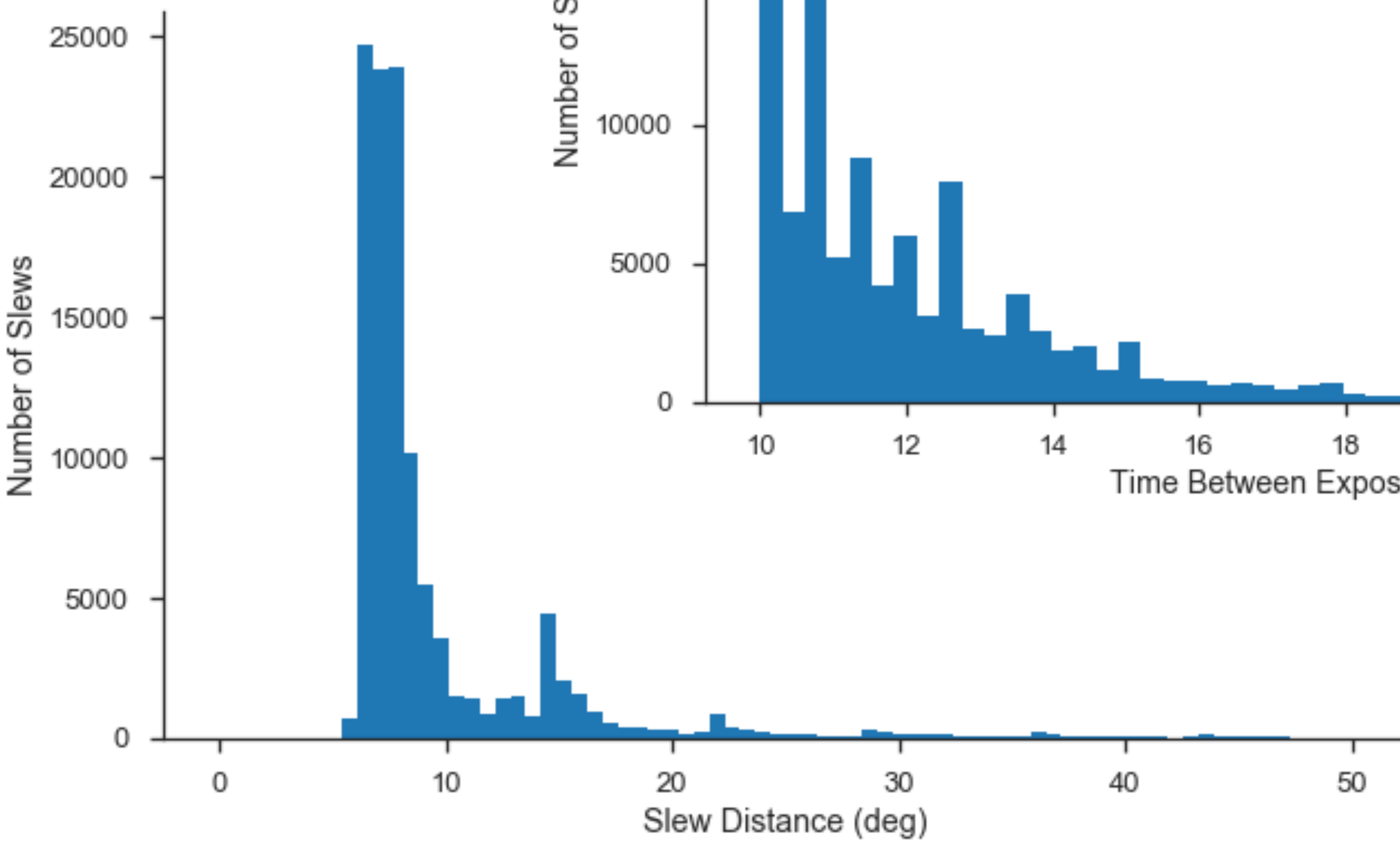


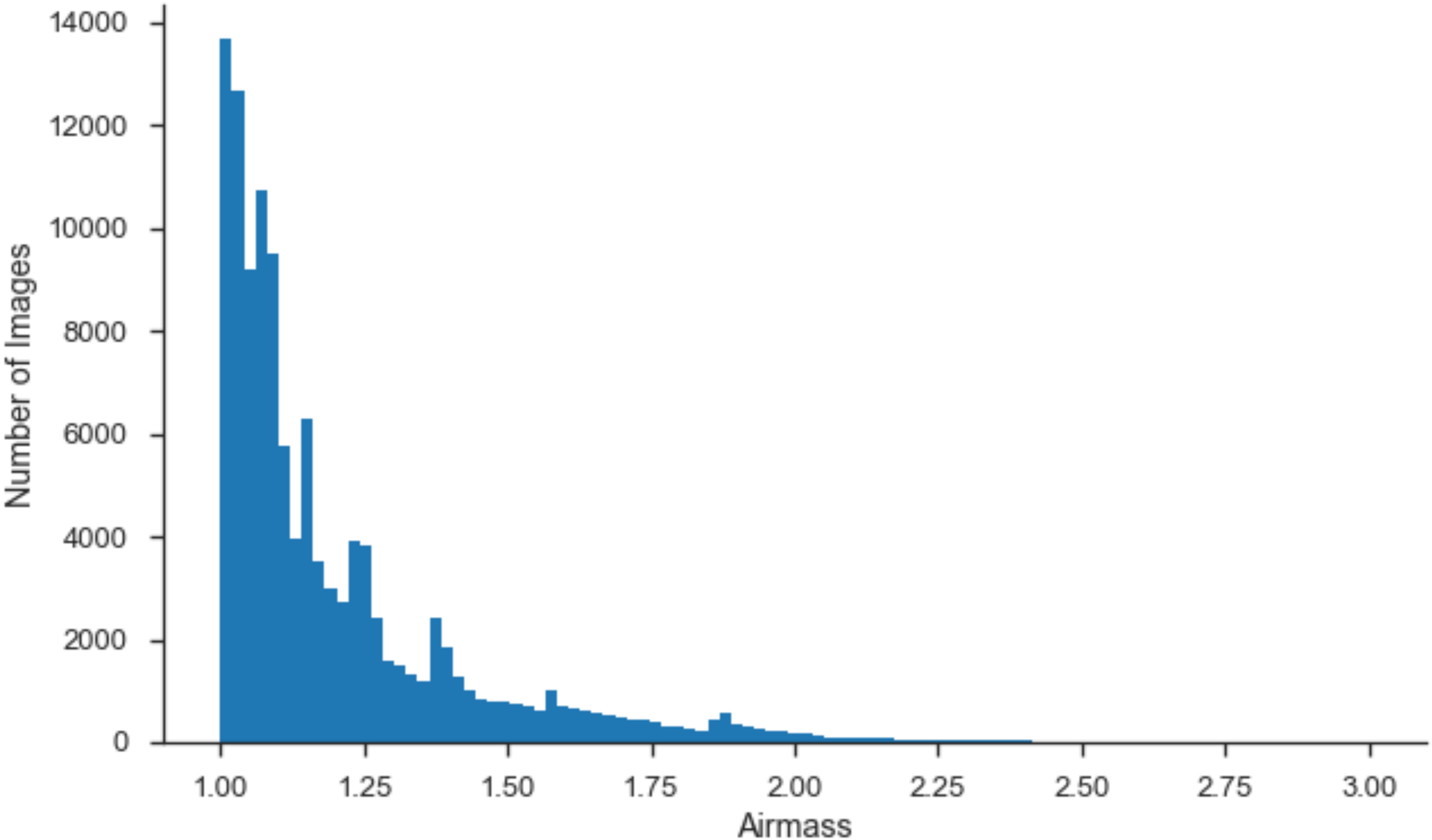

84-90% of sequences completed as requested

# Slew behavior is reasonable; there is room for additional optimization.

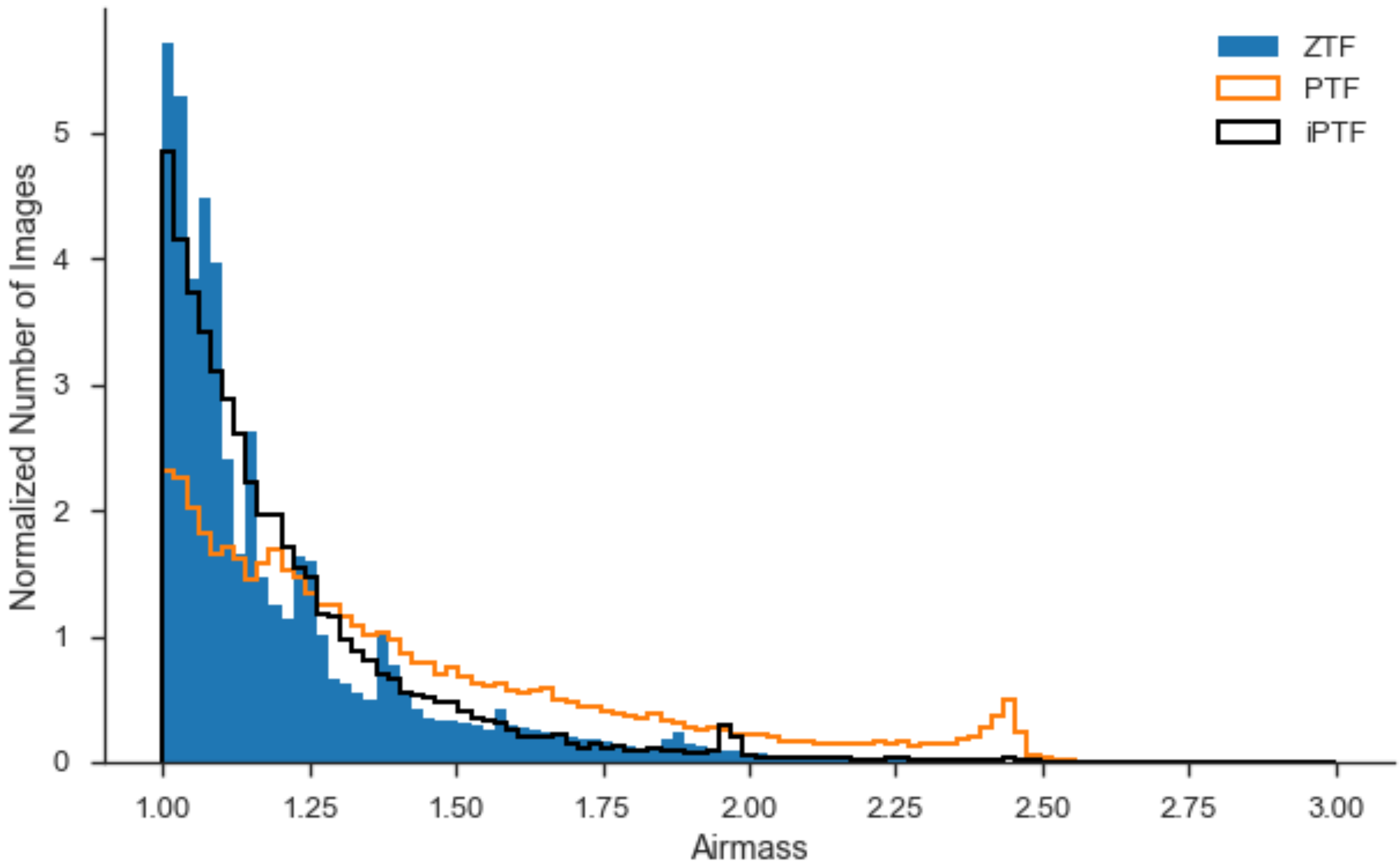


# The scheduler preferentially observes near zenith.



# The scheduler preferentially observes near zenith.



# TOO observations are not yet enabled.

Can combine the volume-probed slot based scheduling with localization probability for each tile

Envision having separate TOO queues that replace the default queue and run until completion within a validity window

Working with TOO marshal team to define the API to talk to the scheduler (which has a REST interface)

## Some necessary work remains.

Integrate TOOs & special programs as swappable block queues

Model as-delivered ZTF performance

Within-night recomputes and other second-order optimizations

Bugfixes and maintenance!

Help with analysis of simulated or actual schedules is welcome.