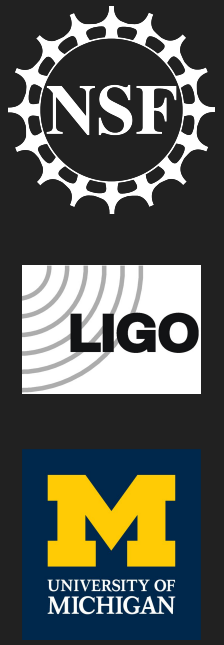


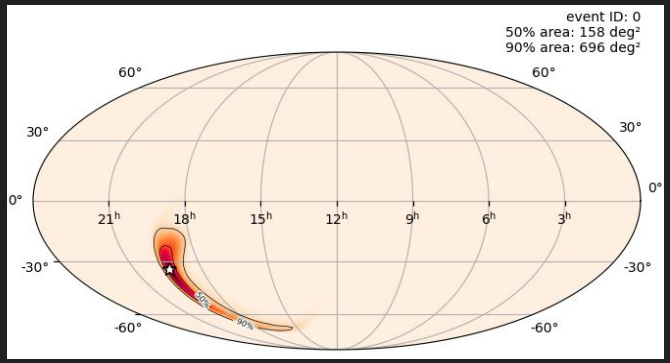
pySLIDE

Python-based Skymap Localization with Inpainted Data Editor

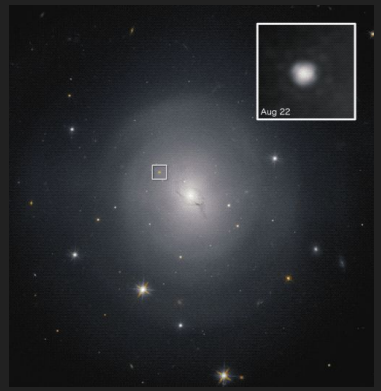
Caltech

LIGO SURF 2021
Maggie Huber
Mentor: Derek Davis

Background



BAYESTAR skymap for one of our simulated GW signals we tested in our PySLIDE workflow.



Hubble optical image of the kilonova produced from binary neutron star merger Gw170817.

Skymaps show us where gravitational waves come from!

And we get cool pictures when we point telescopes there!

To get cool pictures, and do cool multi-messenger astronomy, skymaps need to be:

- Fast
- Accurate (unbiased)

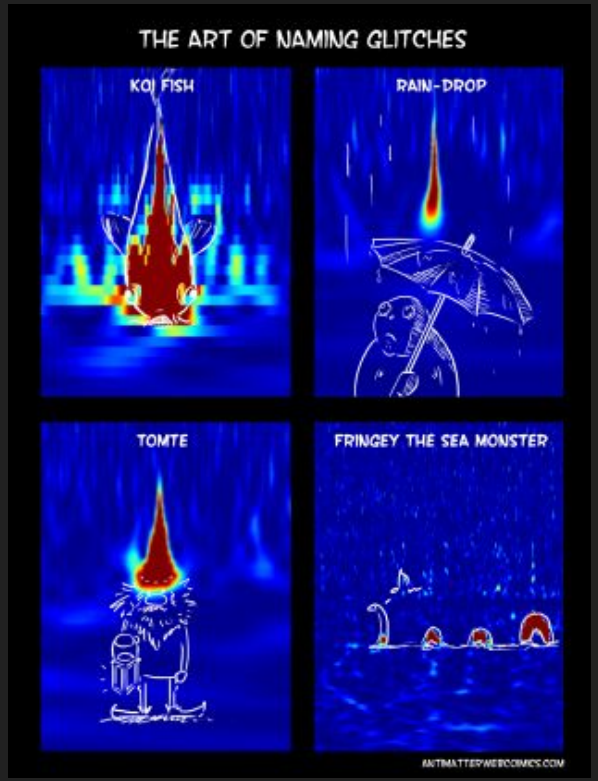
So we don't waste telescope time that can be used for other cool things.

Background

- Motivation for our work: glitches
  - *“Bursts of excess power”*
  - Cause often unknown
  - Many types
  - Some bias sky localization, parameter estimation

- Removing a segment of GW signal with glitch can introduce more bias

- We want to:
  - Correct for this bias by correcting the measured signal-to-noise ratio (SNR)
  - This is called ‘reweighting’ the SNR timeseries



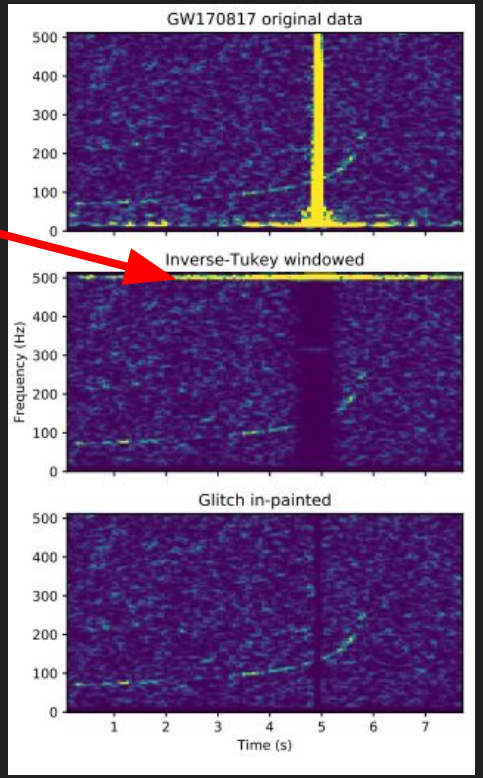
Glitch cartoons from antimatterwebcomics.com

Inpainting

- Generally use *gating* to remove data
  - Smoothy zeroes out bad data
  - This can introduce **excess power leakage**

- We use a method called *inpainting*
  - Exactly masks the bad seconds
    - How? Calculates and subtracts the effect of simply zeroing out
  - Doesn't affect the rest of the data

- Inpainting alone can introduce bias
  - We see this clearly on skymaps when we remove a large time segment



Zackay et al. 2019

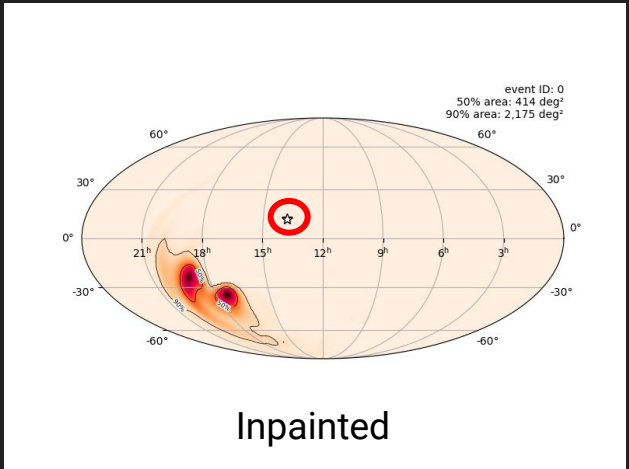
Removal methods demonstrated on BNS event GW170817.

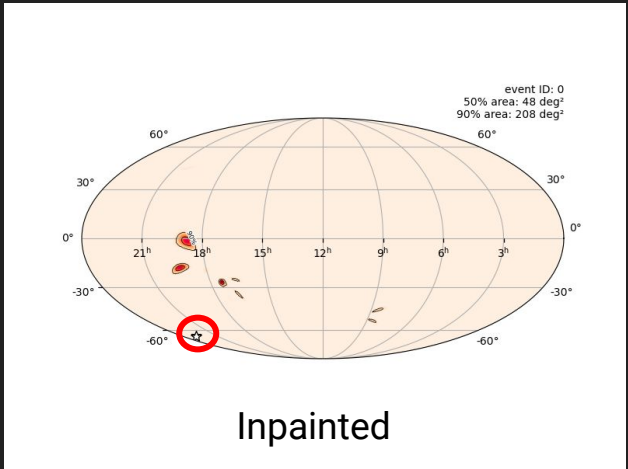
Top panel: raw data with excess power around 5 seconds.

Mid panel: Using normal (Tukey window) gating. Note excess power leakage at the top.

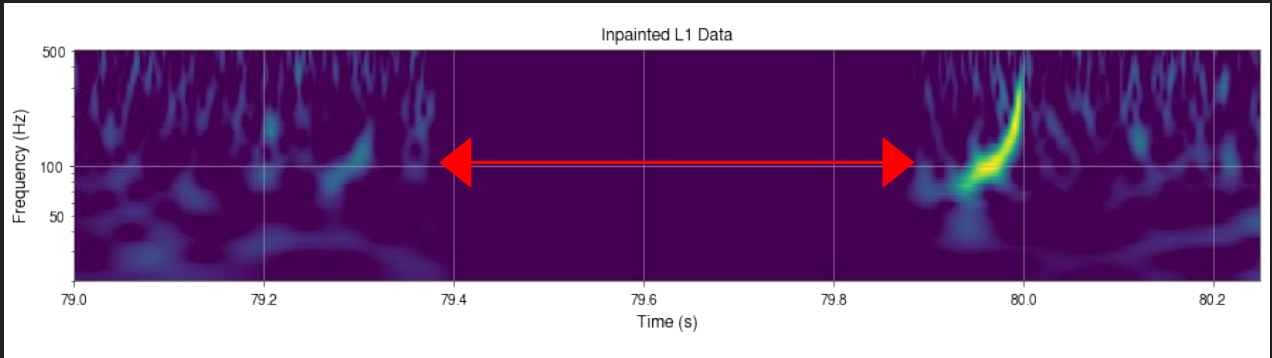
Bottom panel: Using inpainting. More data recovered, and no excess power leaked.

Inpainting bias





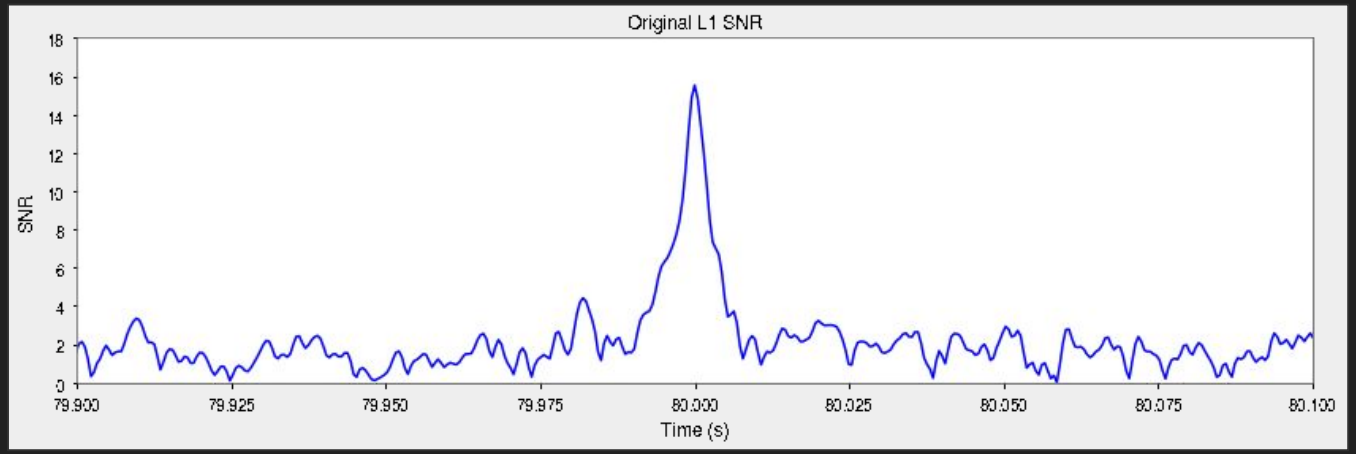
Demonstration of bias on BAYESTAR skymap from removing data. Red circles mark the true location of the signals. The shaded portions are where BAYESTAR selects 50 and 90 percent credible regions for the location of the signal.
Bottom panel: Time-frequency representation of the zeroed portion (red arrows) of a signal.



- When we remove data, we lose information
  - Inaccurate skymaps
- Large gates like these necessary for long-lasting glitch types

Reweighting

- Skymaps need three quantities: time delay, phase, and **amplitude (SNR)**
- A glitch or hole in the data can cause the SNR to deviate from the original amount
- We can go through the data and correct the SNR to increase accuracy of skymap, then renormalize



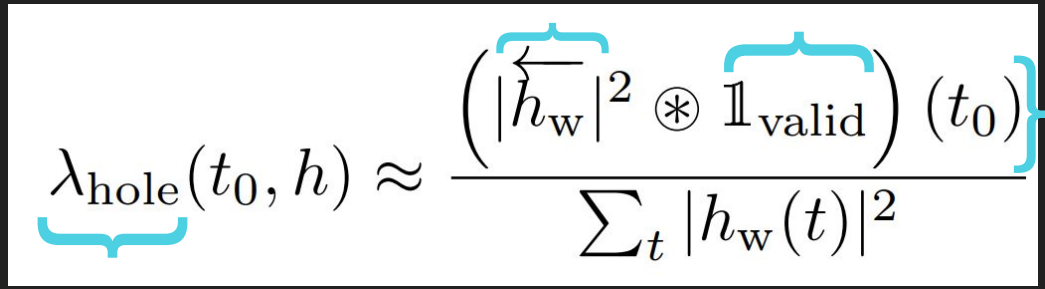
Signal-to-noise ratio timeseries plots for an injected signal. The inpainted timeseries is an inaccurate fit for the original, while the reweighted timeseries is closer to the original.

Reweighting

- How do we actually do it?
  - We find the amount we need to correct the SNR timeseries by running a template through each data point

Template (whitened)

1 when data valid, 0 otherwise



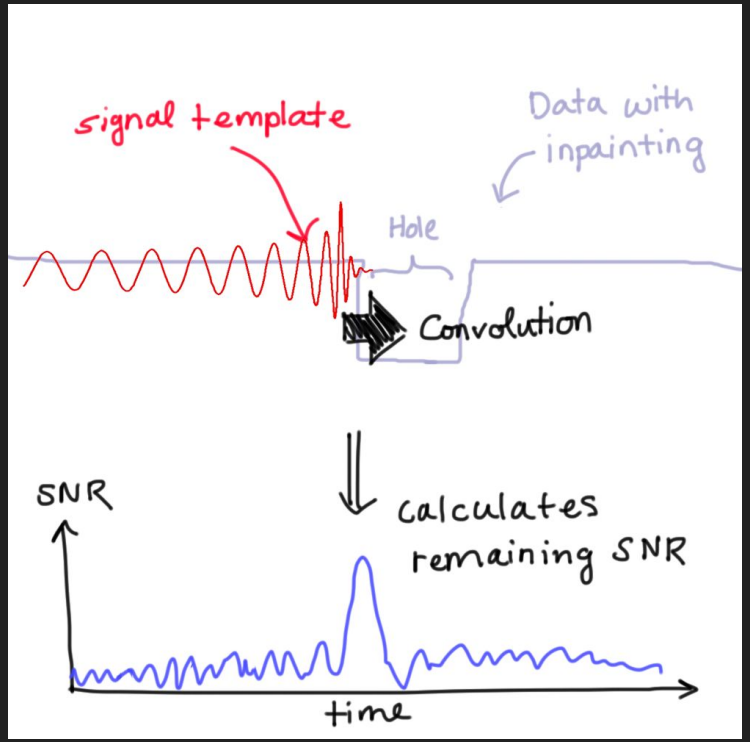
This is a function of merger time

SNR remaining after inpainting

Reproduced from Zackay et al. 2019

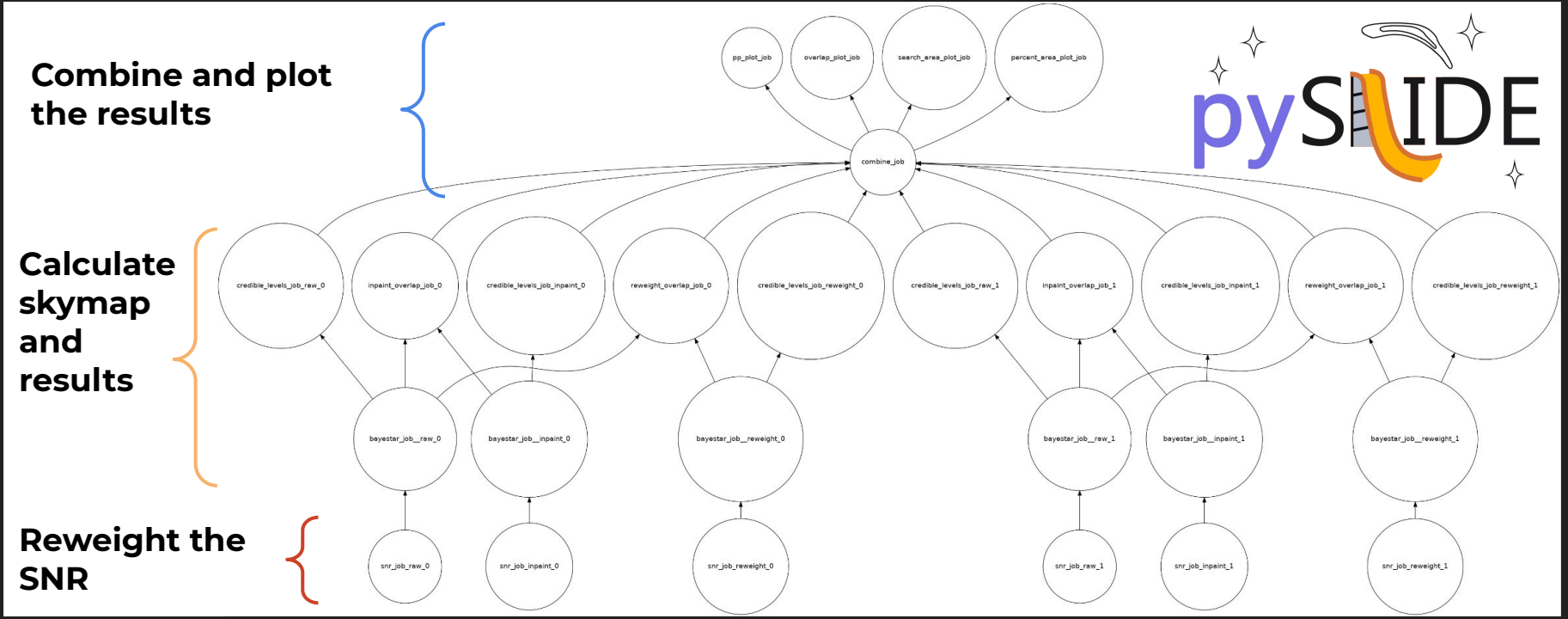
Reweighting

- **Convolve** the template with the “hole” to see where the SNR changes in the inpainted data and correct for it



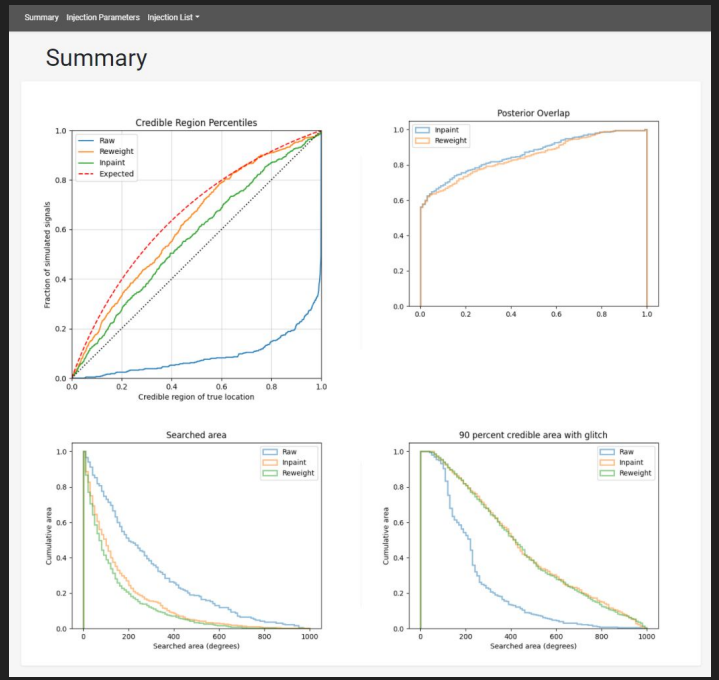
- This method is:
  - Independent of the gate, template
  - Deterministic - one formula
  - Instantaneous

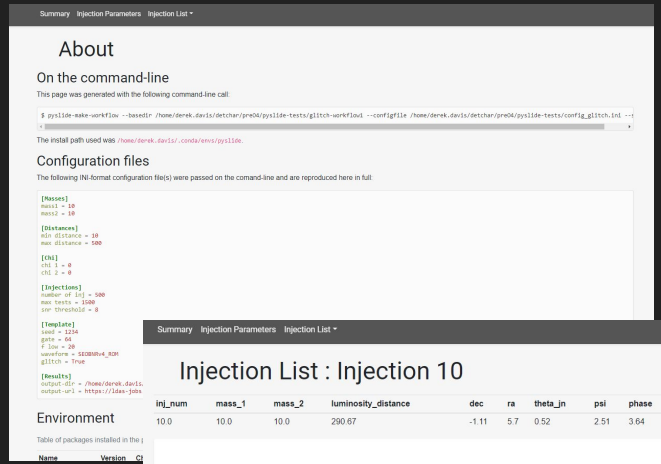
Workflow setup



- We handle all the scripts in a PyCondor workflow run in the LIGO data grid
- Results are displayed as figures showing different metrics

Results of our test

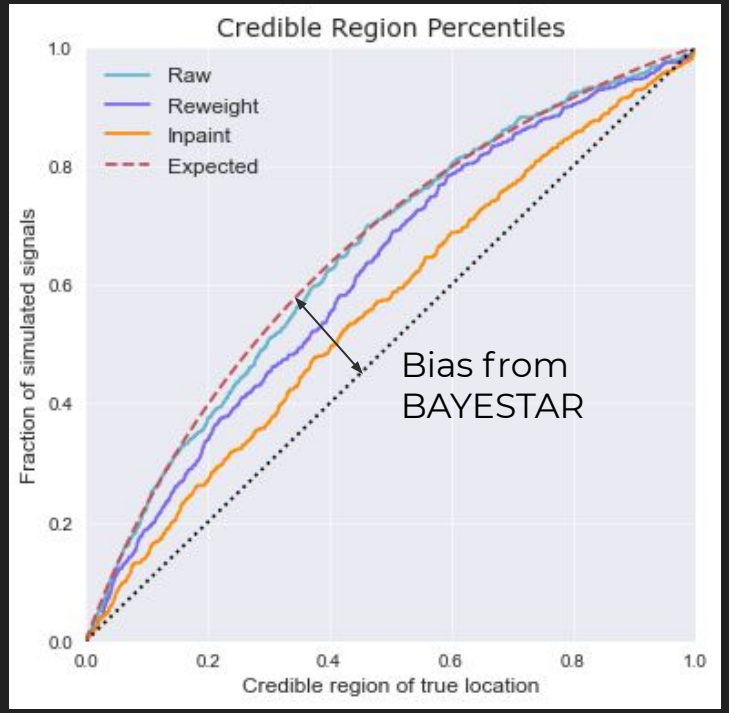


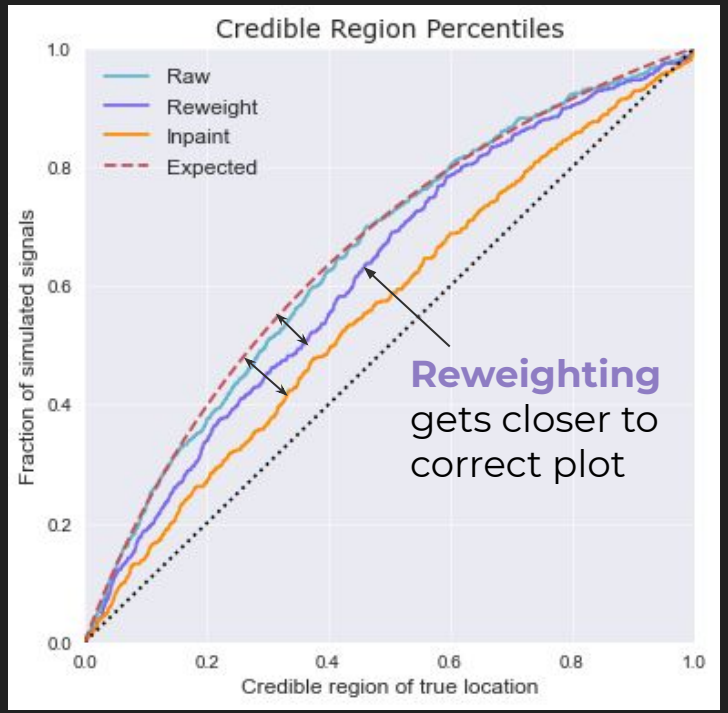


Results pages:

- https://ldas-jobs.ligo.caltech.edu/~derek.davis/detchar/preO4/SURF2021/noglitch-workflow1/
- https://ldas-jobs.ligo.caltech.edu/~derek.davis/detchar/preO4/SURF2021/glitch-workflow1/

Results of our test - P-P plots

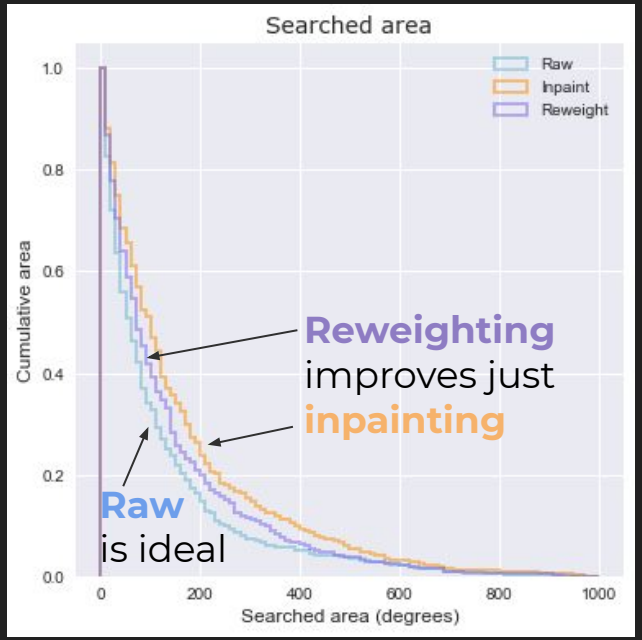




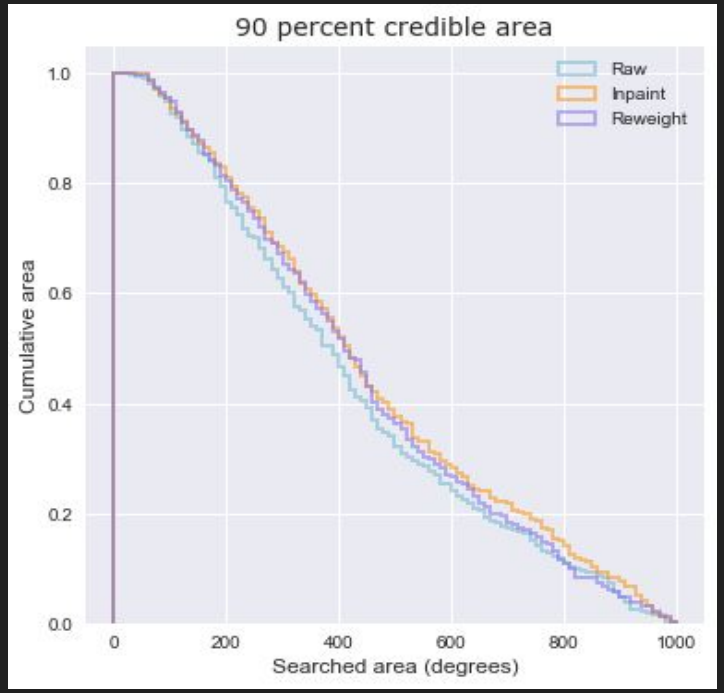
Credible region containing the true localization vs. fraction of signals. Note error overestimation due to BAYESTAR fudge factor. Bayestar rescaling for PyCBC discussed in G2100112

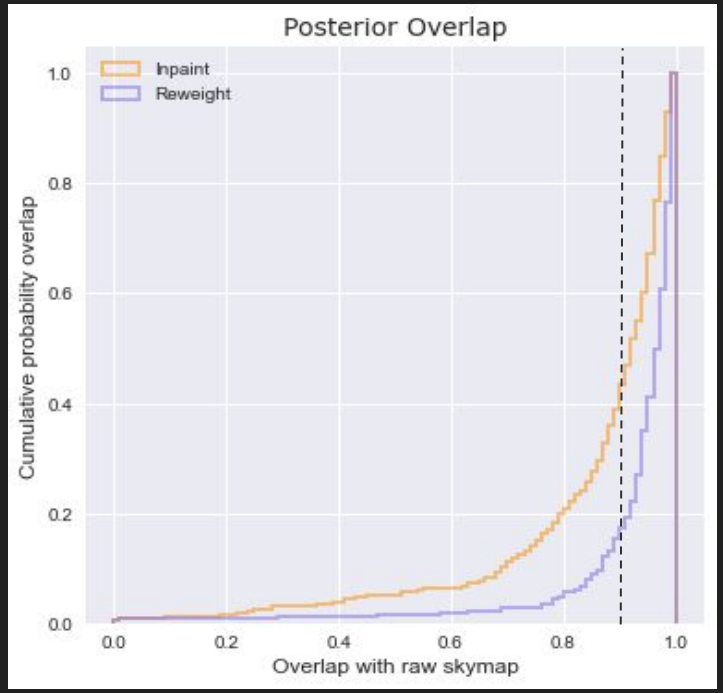
Results of our test - Search area

- The area searched in degrees combined, area of the 90% credible region are more accuracy tests
  - Our method is shown to do better than just inpainting using this test, and only a 10th of a second was removed



Results of our test - continued

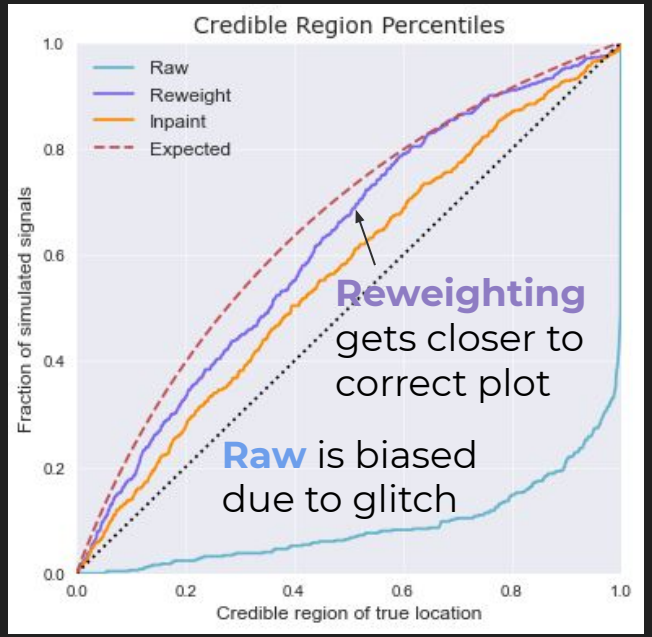


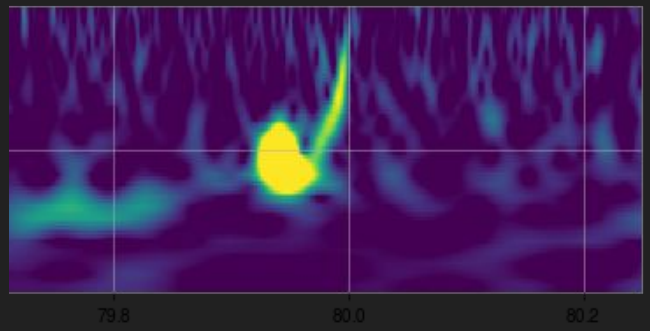


Left: Total area of the 90 percent credible region. The inpainted and raw areas should be the same due to the same renormalization applied to the PSD. Right: Posterior overlap between inpainted and raw data. 90 percent of signals should fall to the right of the 90 percent overlap.

Adding a simulated glitch

- We can inject a glitch into the simulated signal by creating a sine-gaussian wavelet and adding it to the data

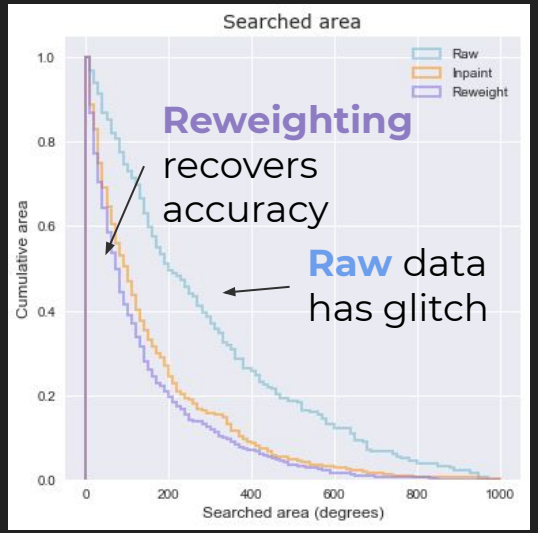


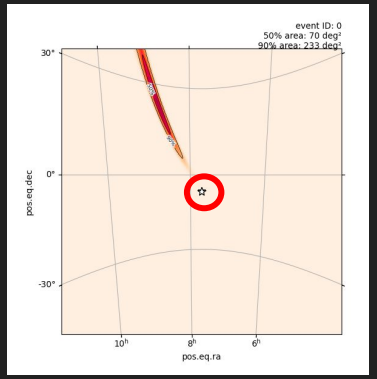


- The glitch we created was found to bias the skymap
  - Reweighting, as seen in the P-P plot, can correct for this

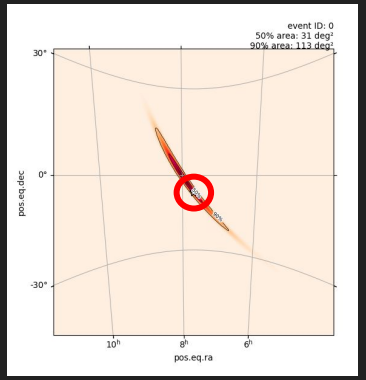
Adding a simulated glitch

- Example skymap above shows how the glitch biases the skymap away from the accurate location
- Searched area and skymaps also verify that reweighting is an effective method for mitigating our simulated glitch

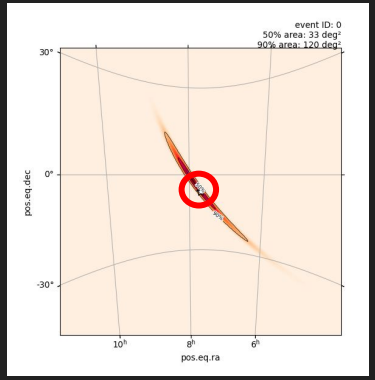




Raw skymap



Inpainted skymap



Reweighted skymap

Summary

- When we remove segments of GW data due to glitches
  - Use **inpainting** to prevent excess power leakage
  - We lose information when inpainting ➡ biases sky localization

- **Reweighting** - new method to account for lost information
  - Shown to improve accuracy of sky localization when tested on large gate widths
  - Also improves accuracy when we add a glitch to the data

- **pySLIDE** will eventually
  - Apply the reweighting method rapidly to both simulated GW strain data, maybe real data in the future
  - Expand to include more features allowing more control over the gate, glitch model, etc.
  - Become open to the public before the next observing run

Acknowledgements

Thank you to the NSF for funding the LIGO SURF program, to Derek Davis for advising and collaborating with me on this project, Alan Weinstein for being Program Director, and to all the students and mentors that made this summer possible.