# Cutting Edge Computing for the Extraction of Astrophysical Parameters from GW Observations

Halston Lim

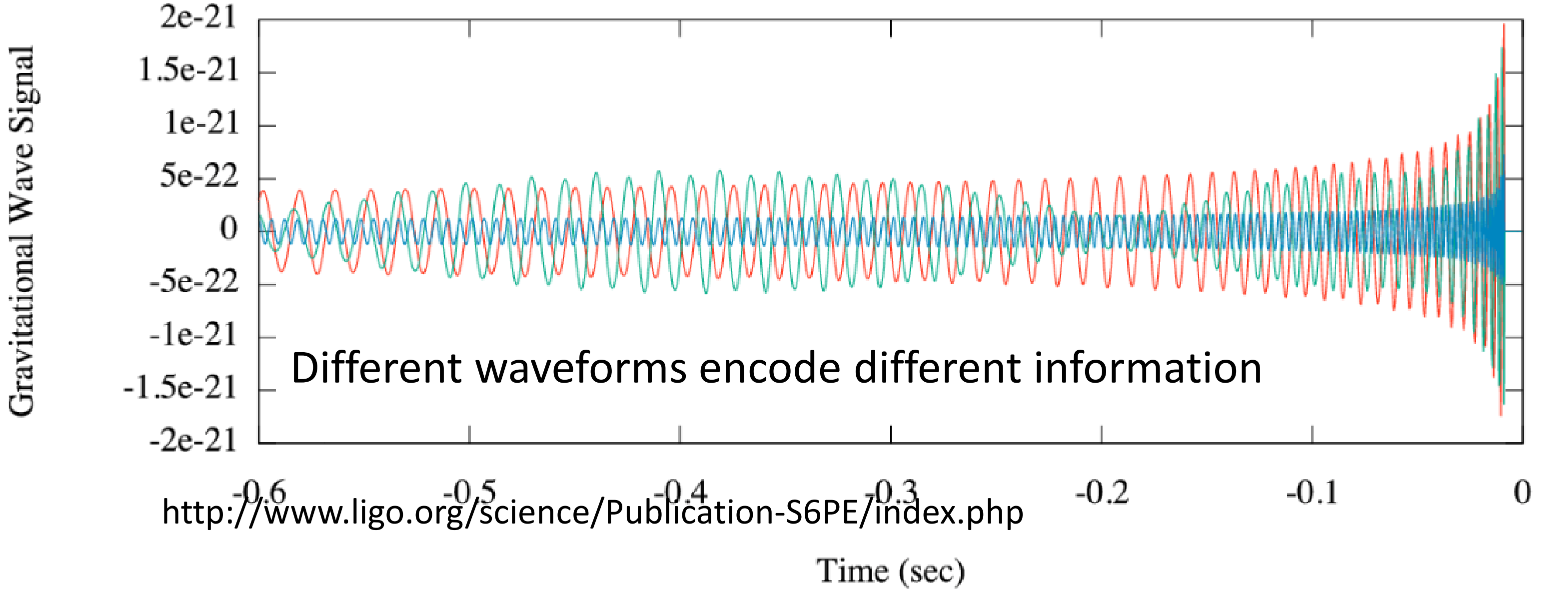
Mentors: Kent Blackburn, Vivien Raymond, Rory Smith

LIGO SURF 2014

# Parameter Estimation

- Process of extracting source physics from GW signals
  - Goal is to estimate the **parameters** of a given waveform model (e.g. mass, orientation, time of calescence, spin)
  - Bayes' Theorem to calculate the posterior probability density (PDF) – the probability of the parameters, given the observed data

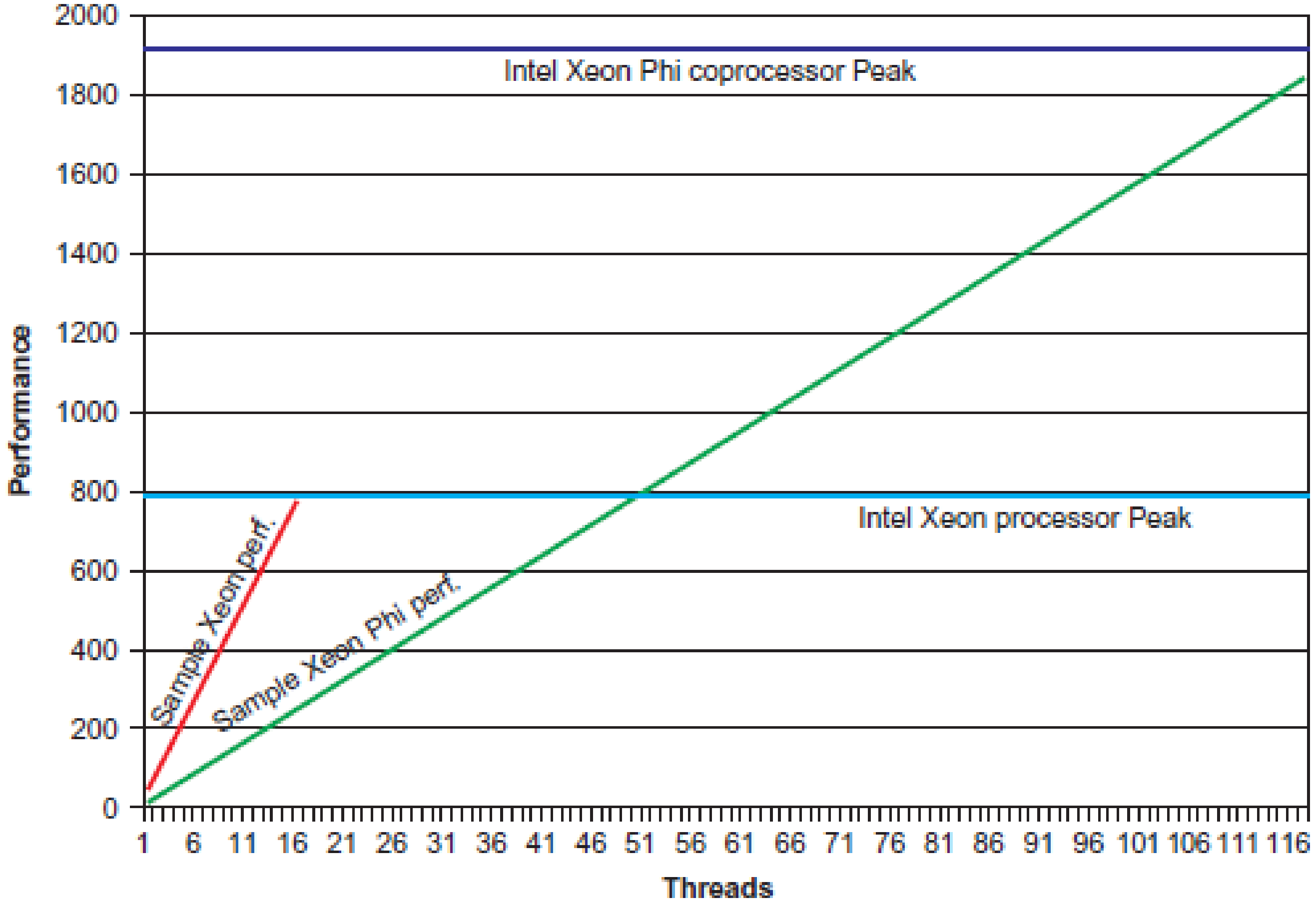
- High computational cost for exhaustive sampling
  - Markov Chain Monte Carlo methods to (more) efficiently calculate PDFs
  - Approximations needed




http://www.ligo.org/science/Publication-S6PE/index.php

# Computer Architecture and Optimization

- Faster processors
  - Intel Sandy Bridge vs. Nehalem
- Parallel architecture – many simultaneous calculations
  - Intel MIC architecture
  - Must write tailored code
- Using functions optimized for specific hardware
  - Intel Math Kernel Library (MKL)




J. Jeffers and J. Reinders, *Intel Xeon Phi Coprocessor High-Performance Programming* (Elsevier, Waltham)

# LIGO Supercomputers

- Ran on the Caltech and Stampede (Texas Advanced Computing Center) clusters
  - High throughput – running many separate jobs (not parallel), CIT
  - High performance – running single job highly parallel, Stampede
  - Different computer usage types give different results

- Computer clusters are shared between many users – jobs are handled through job scheduler (HT Condor, SLURM)

- Stampede, ranked 7th supercomputer according to TOP500 project as of June 2014

# Research Goals

1. Create a benchmark application to test the performance of functions used in GW parameter estimation

2. Optimize the LALSuite parameter estimation code on the Stampede cluster

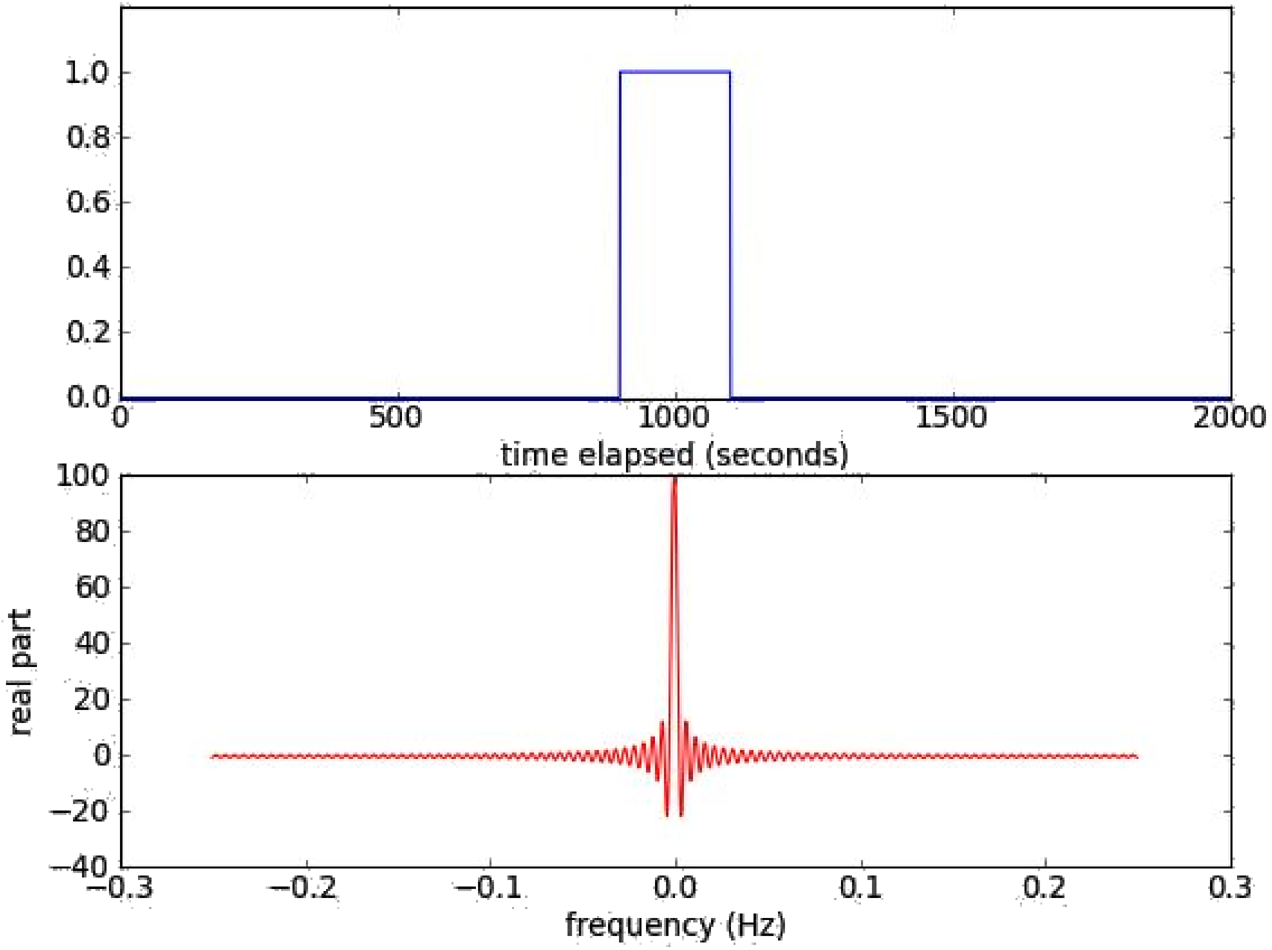
3. Investigate how the Reduced Order Quadrature method speeds up parameter estimation

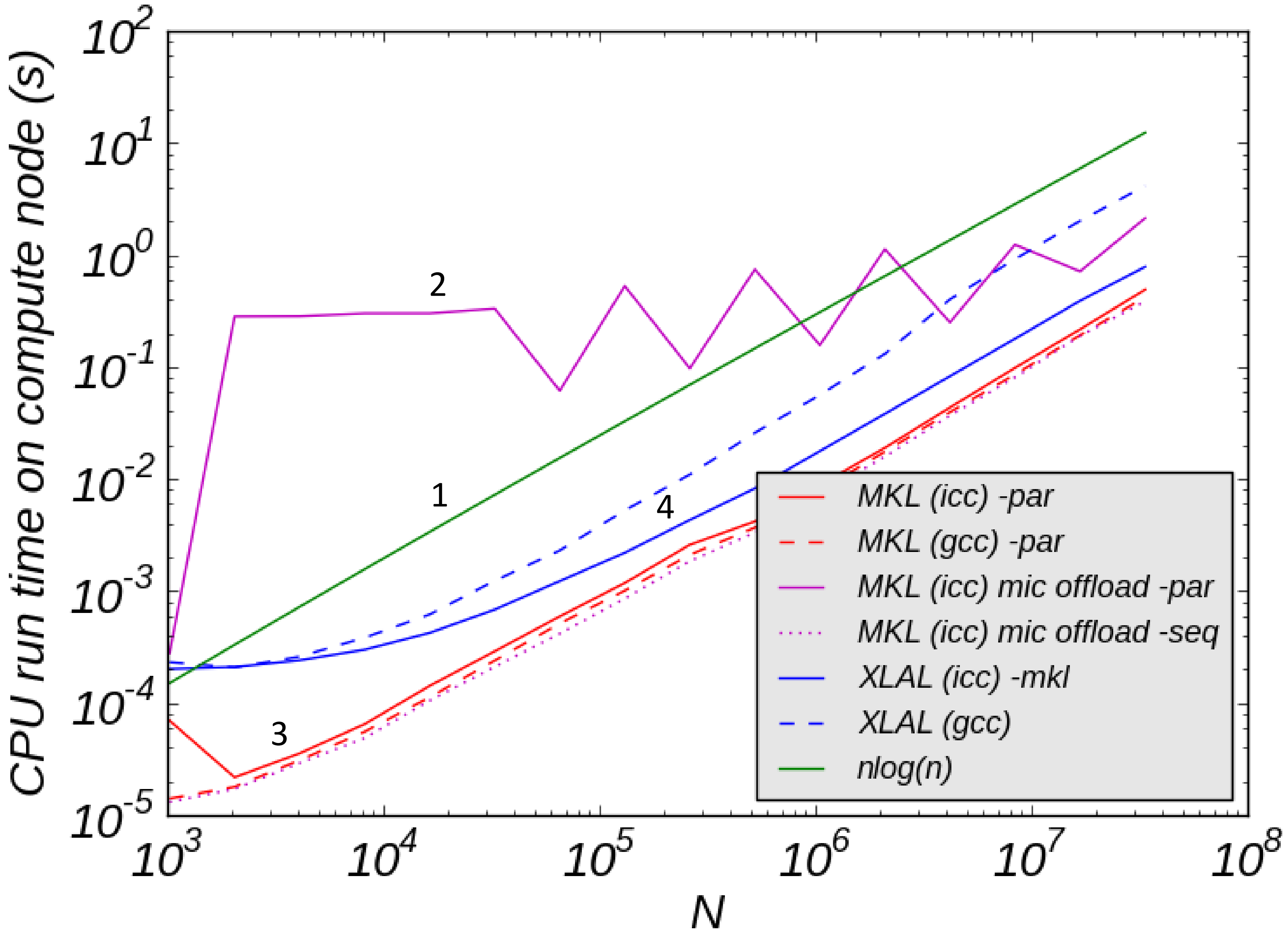
| User | Usage (SUs) |
|---|---|
| Stuart Anderson (stuart) | 1491208.7330 |
| Joseph Areeda (areeda) | 0.0000 |
| Juan Barayoga (barayoga) | 0.0000 |
| James Blackburn (kent) | 74764.7820 |
| Duncan Brown (dabrown) | 172.9350 |
| Santiago Caride (sano) | 0.0000 |
| Marco Cavaglia (cavaglia) | 0.0000 |
| Nelson Christensen (nchriste) | 0.0000 |
| Michael Coughlin (coughlin) | 0.0000 |
| Peter Couvares (pfcouvar) | 2177.6240 |
| Edward Daw (edaw) | 0.0000 |
| Benjamin Farr (bfarr) | 2052.3290 |
| Evan Goetz (egoetz) | 0.0000 |
| Chad Hanna (channa) | 0.0000 |
| Ian Harry (spxiwh) | 0.0000 |
| Paul Hopkins (hopkinsp) | 0.0000 |
| Ra Inta (inta) | 0.0000 |
| Sergey Klimenko (klimenk0) | 0.0000 |
| Dan Kozak (dkozak) | 0.0000 |
| Tjonnie Guang Feng Li (tgfli) | 0.0000 |
| Halston Lim (hblim) | 26078.6330 |

# Benchmark Results

- Create benchmark program to find CPU for a sinusoidal forward FFT

- Benchmarked on the Stampede cluster comparing
  - Intel MKL vs. XLAL (LIGO Algorithm Library) functions
  - Intel compiler vs. generic GNU compiler
  - Threaded vs. sequential MKL
  - MIC  vs. no MICs






1. Theoretical FFT algorithm should scale as nlog(n)

2. MIC processors most scalable, but too much overhead

3. MKL outperformed XLAL by order of magnitude

4. Inefficiencies in the MKL wrapper

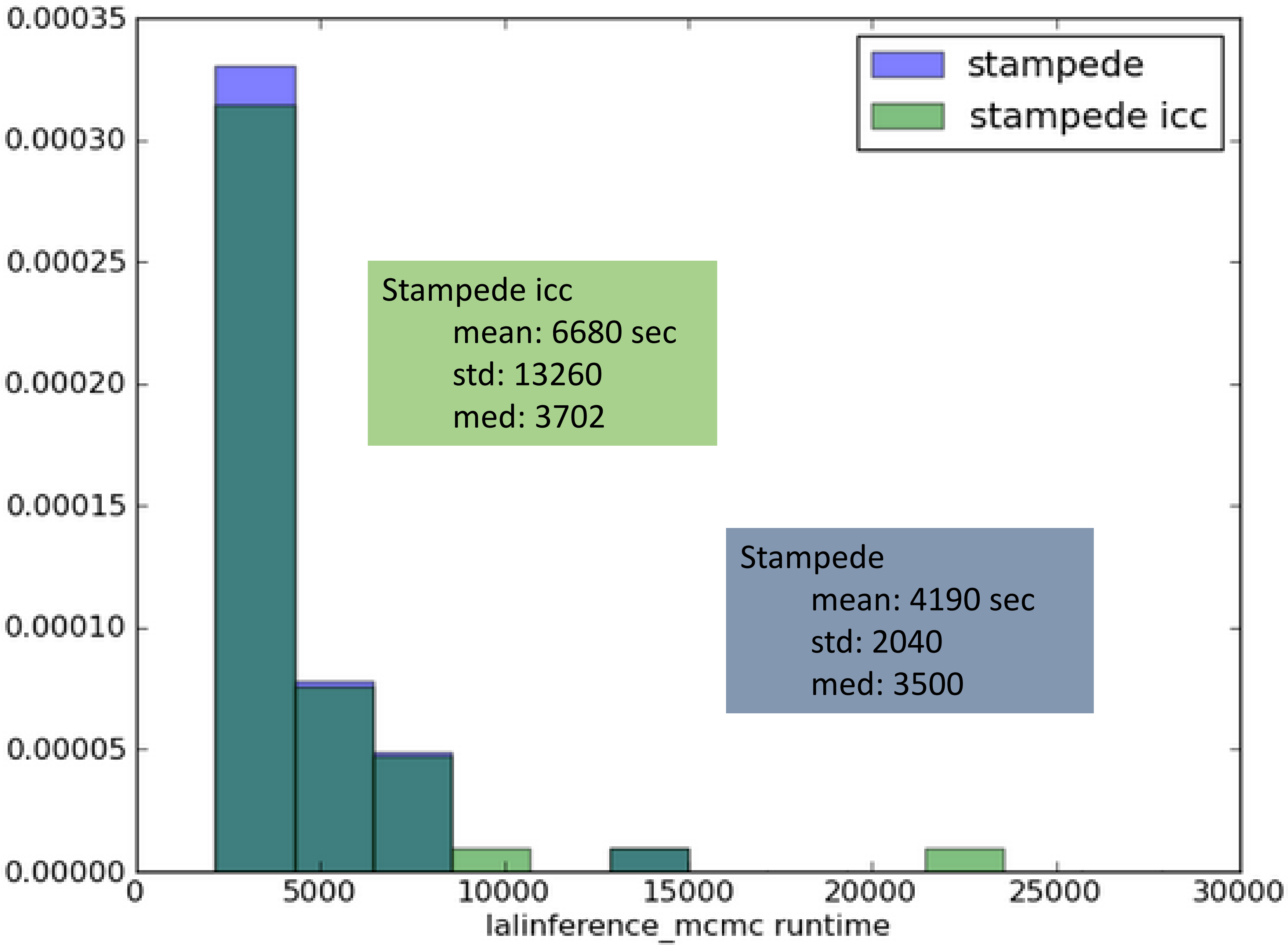
# Comparing Performance with the LALSuite Parameter Estimation Code

- Compared wall-clock time of parameter estimation code with GNU and Intel compiler
  - Simulated data in Advanced LIGO-Virgo detectors
  - Injected 50 binary neutron star signals with fixed component masses of 1.4 $M_\odot$ and uniform in volume and orientation
  - SNR from 5 to 20
  - Taylor F2 3.5 PN (no spin model)

- Parameter estimation code is more than just FFT, so runtime improvements may not be as drastic

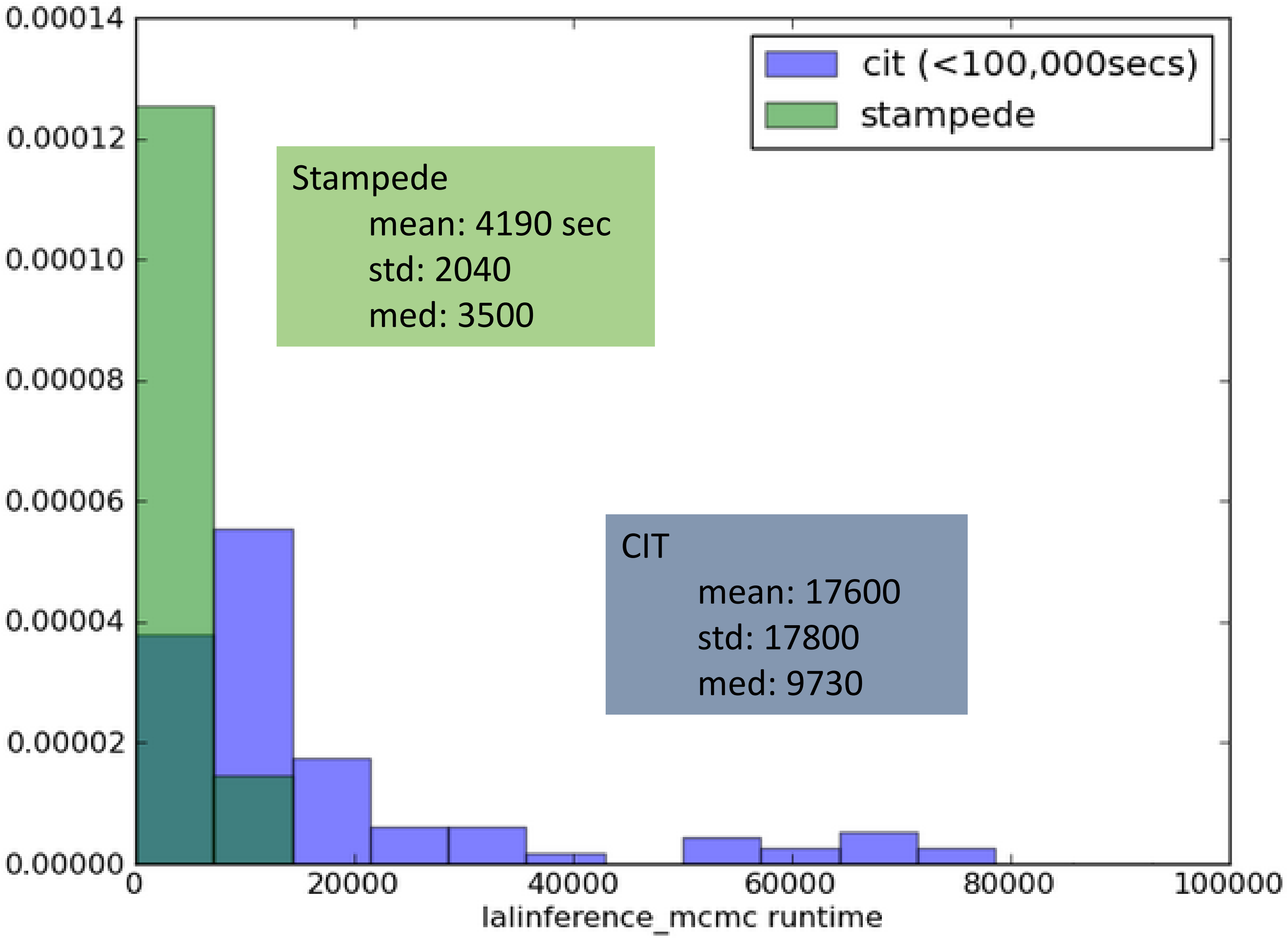
- Runtime differences between ICC and GCC compiler

- Compiler had little impact on speed up
  - In rough agreement with the FFT results



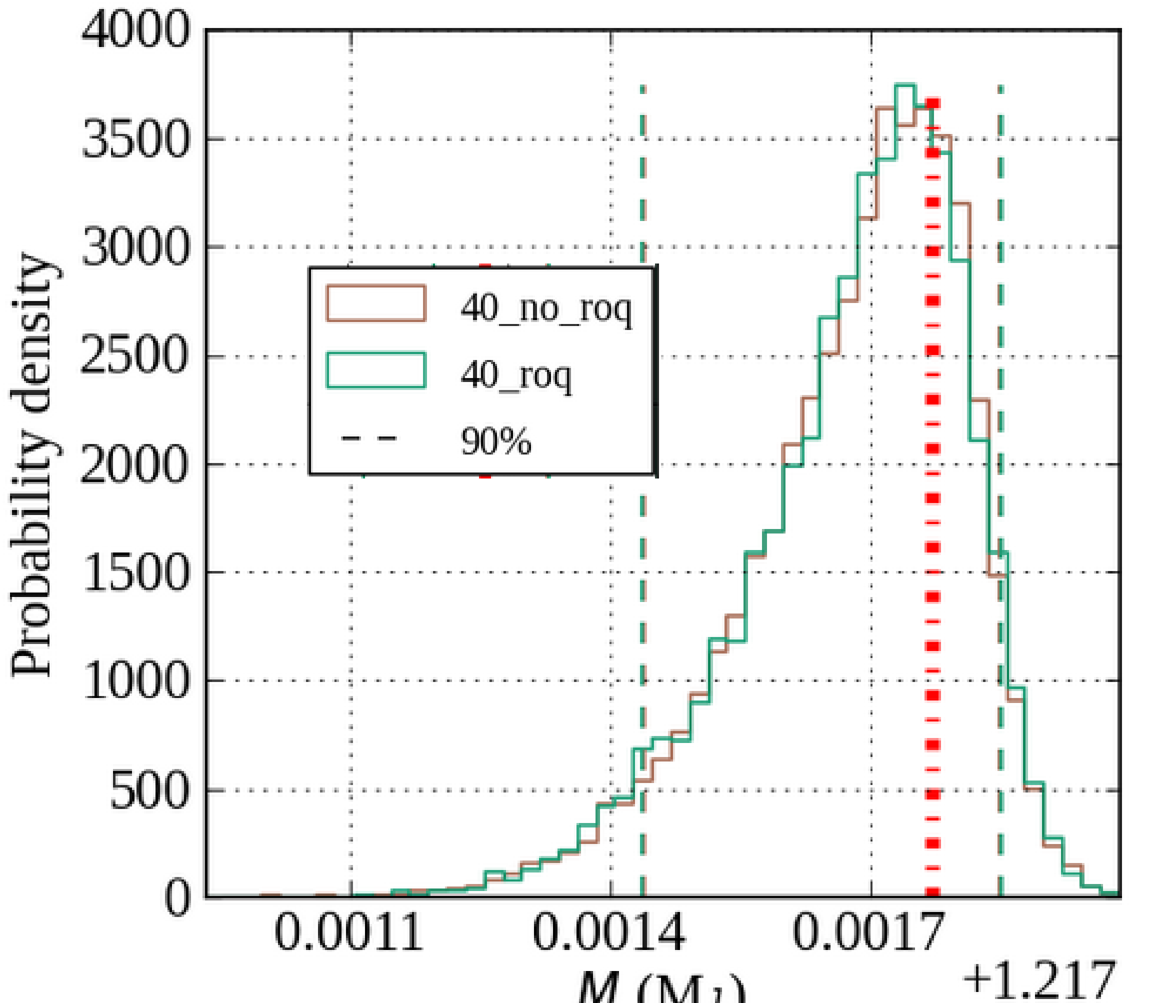
# CIT vs. Stampede performance

- GCC compiled
- Excluded 11 outliers in CIT set that had time > $10^5$ secs
- Stampede runs were both more consistent and efficient



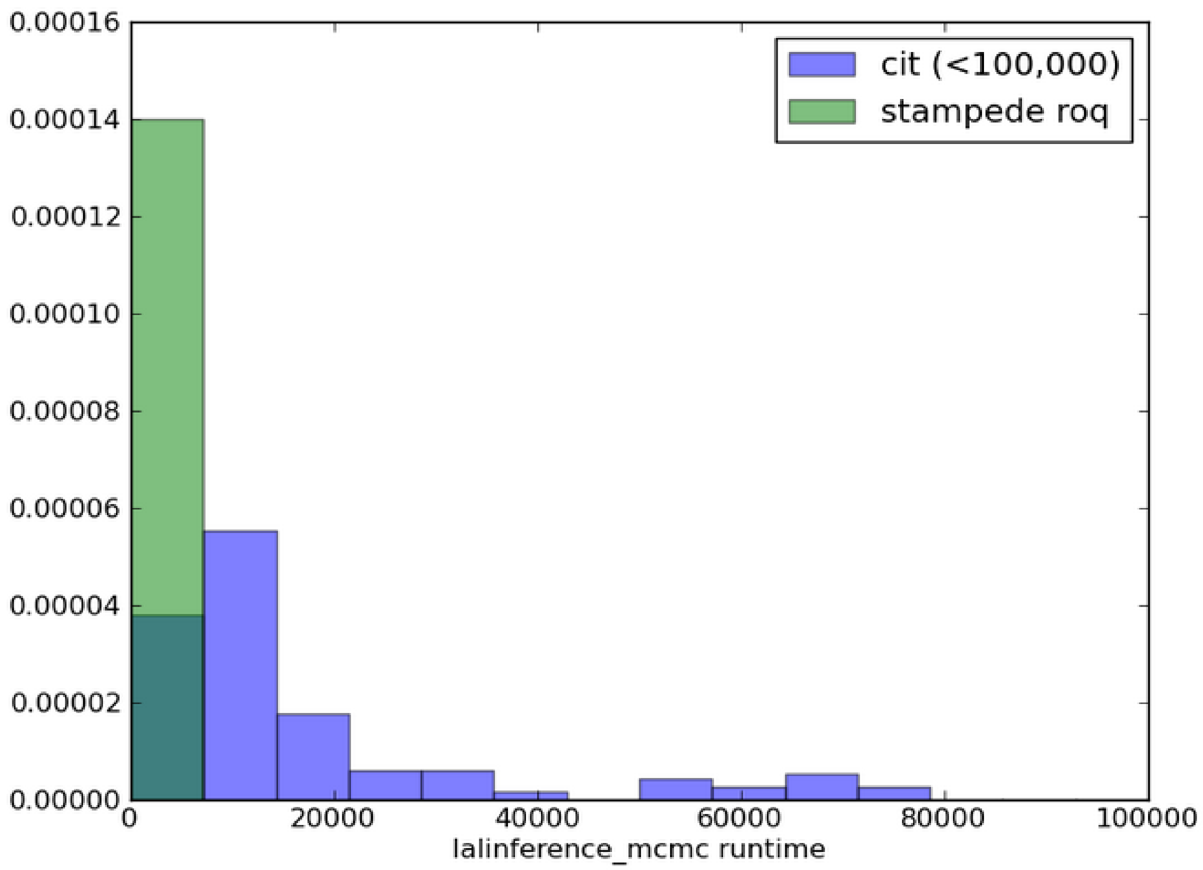
# Incorporating Reduced Order Modeling (ROQ)

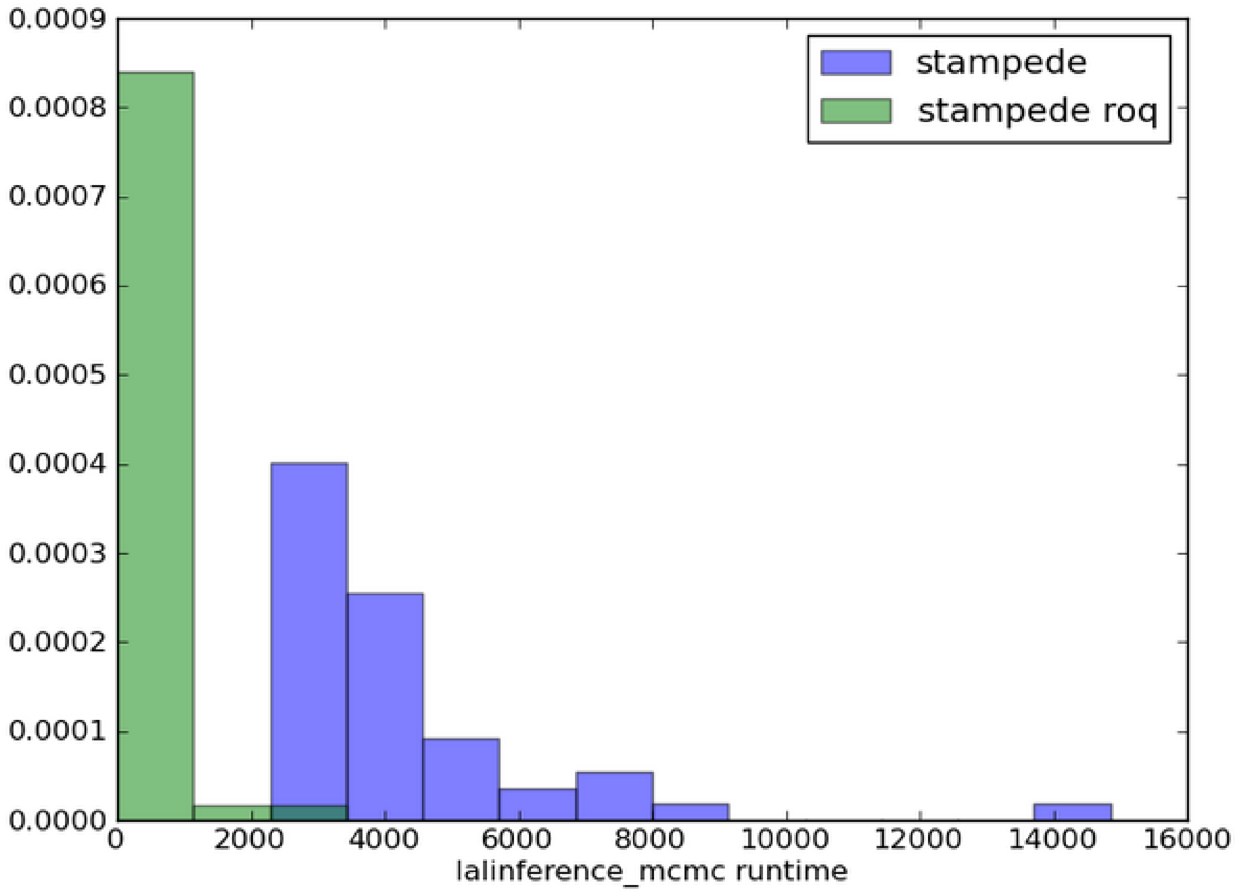
- Reduced order quadrature scheme can reduce parameter estimation times by a factor of 30*
- Faster runtime = more physics
- Comparison for a single injection
  - SNR 17.043
  - **Non-ROQ** 90 minutes to generate
  - **ROQ** 7 minutes to generate




* P. Canizares, S. E. Field, J. Gair, V. Raymond, R. Smith and M. Tiglio, arXiv:1404.6284 [gr-qc].

# ROQ relative performance at $f_{low}$ = 40 Hz



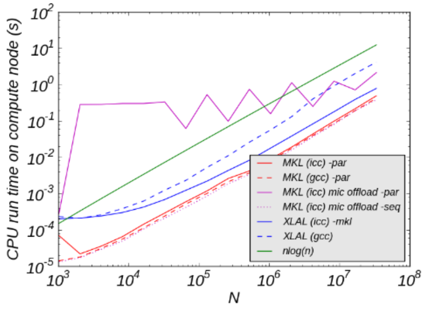

| | Stampede | Stampede ROQ | CIT |
|---|---|---|---|
| Mean CPU time (s) | 4190 | **330** | 17600 |
| Standard Deviation | 2040 | **449** | 17800 |
| Median | 3500 | **218** | 9730 |

# Future Work: Lowering the detection frequency

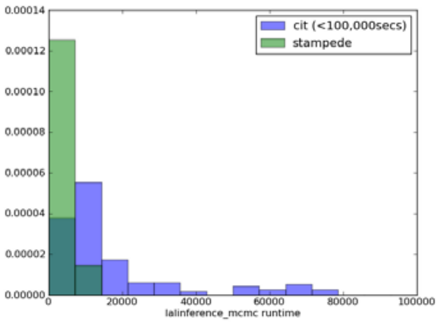
- Lowering detection threshold frequency means more GW signal can be detected
  - Pros: Detecting more signal can give more information about source -> better parameter estimation
  - Cons: More computational intensive to analyze
- Can we justify lowering f_low through a parameter estimation improvement?
- How can ROQ enable us to answer this question:

| Method Name | 40 Hz Analysis Time | 20 Hz Analysis Time |
|---|---|---|
| Non-ROQ | 90 minutes | ~1 week |
| ROQ | 7 minutes | 60 minutes |

# LIGO Conclusion




Various algorithms used in GW analysis like the FFT can be optimized on hardware




Large speed ups possible for parameter estimation code

- Running on HPC nodes such as Stampede (3-5 X faster)

| Method Name | 40 Hz Analysis Time | 20 Hz Analysis Time |
|---|---|---|
| Non-ROQ | 90 minutes | ~1 week |
| ROQ | 7 minutes | 60 minutes |

ROQ drastically speeds up parameter estimation

- May make analyzing signals from 20 Hz feasible

# Next steps

- Confirm results at 20 Hz with ROQ
- Work to further optimize LALSuite
  - Incorporate MKL as much as possible, as order of magnitude improvement in FFTs not seen yet in lalinference_mcmc
  - Make applications scalable for the MIC coprocessors
- Test speed ups with spin models at 30 Hz
- Profile the parameter estimation application to reveal inefficient hotspots in the code

# Acknowledgements


- Kent Blackburn, Vivien Raymond and Rory Smith

- Alan Weinstein

- NSF Foundation