Gravitational Wave Parameter Estimation with Compressed Likelihood Evaluations

Scott Field

Department of Mathematics, University of Massachusetts Dartmouth

Workshop III: Source inference and PE in Gravitational Wave Astronomy

November 17, 2021

Simulating eXtreme Spacetimes

Black holes, neutron stars, and beyond...

How to accelerate GW inference with reduced order quadratures?

How to accelerate GW inference with reduced order quadratures?

- Harbir Antil, SF, Frank Herrmann, Ricardo Nochetto, Manuel Tiglio. "Two-step greedy algorithm for reduced order quadratures" (J. of Scientific Computing, 2013)
- Priscilla Canizares, SF, Jonathan Gair, Manuel Tiglio, "Gravitational wave parameter estimation with compressed likelihood evaluations" (PRD, 2013)
- Priscilla Canizares, SF, Jonathan Gair, Vivien Raymond, Rory Smith, Manuel Tiglio. "Accelerated gravitational-wave parameter estimation with reduced order modeling" (PRL, 2015)
- Rory Smith, SF, Kent Blackburn, Carl-Johan Haster, Michael Purrer, Vivien Raymond, Patricia Schmidt, "Fast and Accurate Inference on Gravitational Waves from Precessing Compact Binaries" (PRD 94, 044031, 2016)
- Harbir Antil, Dangxing Chen, SF, "A Note on QR-Based Model Reduction: Algorithm, Software, and Gravitational Wave Applications" (IEEE Computing in Science & Engineering, 2018)
- Jeroen Meidam, et al. "Parametrized tests of the strong-field dynamics of general relativity using gravitational wave signals from coalescing binary black holes: Fast likelihood calculations and sensitivity of the method" (PRD 2018).
- Rory Smith, et al. "Bayesian inference for gravitational waves from binary neutron star mergers in third-generation observatories Authors" (PRL 2021)

Outline

Outline

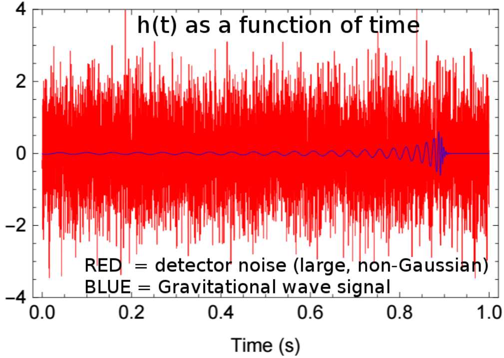
Gravitational wave datasets

- In absence of GWs the distance between two mirrors is L ($\approx$ 4Km)
- GW $h(t)$ causes small, time-dependent ΔL change in length: $h(t) \propto \frac{\Delta L}{L} \leq 10^{-20}$

Time series data recorded as

$$d(t_i) = h(t_i) + n(t_i) \,,$$

Here, d is the data, h is the gravitational-wave signal, and n is the detector noise.



Once a gravitational wave signal has been observed...

Parameter inference: what kind of binary black hole system genereated this signal (masses, spins, sky location, etc...)

Bayesian inference of GW datasets

Assume general relativity correctly models our GW signal $h^{\mathrm{GR}}(t;\mu)$.

- μ is a (15 dimensional) parameter vector
- d is the dataset $d =$ detector noise $+$ gw signal
- $P(\mu \mid d, h^{\mathrm{GR}}) =$ probability of μ *given* observation d and model h^{GR}

Inference problem for GWs (Bayes' theorem)

Having measured d and assuming h^{GR}, compute

$$P(\mu \mid d, h^{\mathrm{GR}}) = \frac{P(d \mid \mu, h^{\mathrm{GR}}) P(\mu \mid h^{\mathrm{GR}})}{P(d \mid h^{\mathrm{GR}})}$$

The prior distribution is constrained by general relativity and informed by astrophysics

Bayesian inference of GW datasets

Assume general relativity correctly models our GW signal $h^{\mathrm{GR}}(t;\mu)$.

- μ is a (15 dimensional) parameter vector
- d is the dataset $d =$ detector noise + gw signal
- $P(\mu \mid d, h^{\mathrm{GR}}) =$ probability of μ *given* observation d and model h^{GR}

Prior probability

Likelihood function

Inference problem for GWs (Bayes' theorem)

Having measured d and assuming h^{GR}, compute

Evidence (normalization)

Posterior probability

$$P(\mu \mid d, h^{\mathrm{GR}}) = \frac{P(d \mid \mu, h^{\mathrm{GR}})P(\mu \mid h^{\mathrm{GR}})}{P(d \mid h^{\mathrm{GR}})}$$

The prior distribution is constrained by general relativity and informed by astrophysics

The likelihood function

Let the dataset be given by

$$d(f_k) = h(f_k; \mu) + n(f_k)$$

at frequencies $f_k = f_1 + (k-1)\Delta f$ with $k = 1, \ldots, N$.

The likelihood function, assuming a Gaussian noise model

$$P(d \mid \mu, h^{\mathrm{GR}}) \propto \exp\left(-\frac{1}{2} \sum_{k=1}^{N} \frac{\left| d(f_k) - h_\mu(f_k) \right|^2}{\sigma_k^2} \right)$$

where the variance σ_k (power spectral density) is determined experimentally from the noise $n(f_k)$

Likelihood computations are too slow

Parameter estimation cost is dominated by $$\sum_{k=1}^{N} \frac{[d(f_k) - h_\mu(f_k)]^2}{\sigma_k^2}$$

Cost will quickly escalate

- If evaluation at a single parameter and frequency value takes $\sim 10^{-6}$s
- Long BNS signals will have $N = 4096 * 64$
  - Notice the cost scales linearly with N
- A typical parameter estimation study has $\approx 10^6$ likelihood evaluations
- Implies 3 days of runtime!! (if done sequentially)

Parameter estimation challenges

1. Bayesian inference: The analysis ranges from frustratingly slow to prohibitively slow
2. Closed-form models: many days to $>$ 100 years (BNS with 3G)
3. ODE models: 1 months to impossible
4. PDE models: cannot do this directly

Approaches to faster PE (non-exhaustive list)

- Make the waveform model faster: surrogate models, feed-forward network models, Phenom* family
- Make the sampling faster: parallelized nested sampling (pBilby), scalable inference (Dan Foreman-Mackey's talk)
- Likelihood-free methods: neural networks with normalizing flows (Stephen Green's talk)
- Make the likelihood evaluation faster: Heterodyned Likelihood/relative binning, multi-band interpolation, reduced-order quadratures
- Use better hardware: ILE/RIFT (GPU-acceleration), pBilby

Reduced-order quadratures (ROQs) in use

ROQs have been used in many of the LVK's gravitational-wave parameter estimation studies. For example:

- GW170817: observation of gravitational waves from a binary neutron star inspiral
- GW170104: observation of a 50-solar-mass binary black hole coalescence at redshift 0.2
- GW170814: a three-detector observation of gravitational waves from a binary black hole coalescence

Benefits/Observations

- The first results are produced with the ROQ-accelerated code (low-latency)
- No issues with non-Gaussian and/or large noise sources (other than likelihood assumption)
- Already available in LALInference, Bilby, parallel Bilby
- We'll come back to the drawbacks later

Outline

Numerical integration (quadrature)

Formulas for numerical integration of a function, $f(t)$, can be written as

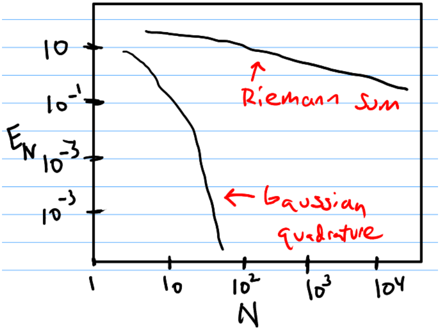
$$\int f(t)dt \approx \sum_{k=1}^{N} \omega_k f(t_k)$$

and the error is

$$E_N = \left| \int f(t)dt - \sum_{k=1}^{N} \omega_k f(t_k) \right|$$

Quadrature rule is defined by a set of weights, ω_k, and points, t_k.

Numerical integration (quadrature)



$$E_N = \left| \int f(t)dt - \sum_{k=1}^{N} \omega_k f(t_k) \right|$$

Examples

- Low-order Riemann sum: $\omega_k = \Delta t$, $t_k = 0, \Delta t, 2\Delta t, \ldots$
  - $E_N \propto N^{-1}$ ...need to take N large for accuracy
- High-order Gaussian quadrature: ω_k and t_k have special values
  - For smooth functions, converges exponentially to the true value
  - $E_N \propto \exp(-N)$ ...small N is still very accurate

Do I need a low-order quadrature rule for noisy data?

Define a weighted inner product between vectors $f, g \in \mathbb{C}^N$ as

$$\langle f, g \rangle := \sum_{k=1}^{N} \frac{1}{\sigma_k^2} f_k g_k^* \,.$$

Then

$$\sum_{k=1}^{N} \frac{[d(f_k) - h_\mu(f_k)]^2}{\sigma_k^2} = \langle d - h_\mu, d - h_\mu \rangle = \langle d, d \rangle + \langle h_\mu, h_\mu \rangle - 2\Re \langle d, h_\mu \rangle$$

Observations

- This is a low-order quadrature rule for computing inner products
- Since h is a smooth function, if we were free to choose the nodes and ignore d, we would have selected a high-order quadrature rule for the last two terms

Do I need a low-order quadrature for noisy data? A: No

$$\langle d - h_\mu, d - h_\mu \rangle = \langle d, d \rangle + \langle h_\mu, h_\mu \rangle - 2\Re\langle d, h_\mu \rangle$$

- $\langle d, d \rangle$ computed once – the dataset is fixed
- Build Gaussian quadrature like-rules for $\langle h_\mu, h_\mu \rangle$ and $\langle d, h_\mu \rangle$

Goal: $$\langle d, h_\mu \rangle = \Delta f \sum_{i=1}^{N} \frac{d(f_i) h_\mu^*(f_i)}{\sigma_i^2} \approx \sum_{i=1}^{n} \omega_i h_\mu^*(F_i) = \langle d, h_\mu \rangle_{\text{ROQ}}$$

- Data-specific weights, ω_i, which depend on the dataset, d, and properties of the detector noise σ_i^2
- Model-specific quadrature nodes $\{F_i\}_{i=1}^{n}$ selected as a subset of $\{f_i\}_{i=1}^{N}$
- N is a property of the experiment whereas n is a property of the model
- Model's approximation properties are *independent* of data, $n \ll N$.

We refer to this dimensionally reduced quadrature as a *reduced order quadrature* (ROQ) rule.

Problem Formulation

Parametrized Model

- Let
$$\mathcal{F} := \{h_\mu : \Omega \to \mathbb{C} \mid \mu \in \mathcal{P}\}$$
be a set of functions where Ω, $\mathcal{P}$ denote the physical and parameter domains.
- Example: h_μ is some GW model, $\mathcal{P}$ are masses/spins for BBH systems, $\Omega = [20, 4096]$Hz.

ROQ roadmap

1. (Offline) Find an n-dimensional approximation space "$X_n \approx \mathcal{F}$"
2. (Offline) Find n points for accurate and stable integration in X_n
3. (Start-up) When data is known compute quadrature weights $\{\omega_i\}_{i=1}^n$
4. (Online) Use new integration rule $\{f_i, \Delta f\}_{i=1}^N \to \{F_i, \omega_i\}_{i=1}^n$

Step 1: Compressing the model

- Seek a representation of the gravitational wave model

$$h_\mu(f) \approx \sum_{i=1}^{n} c_i(\mu) e_i(f)$$

  for n as small as possible

- Sometimes referred to as a *reduced order model* for a special choice of e_i

Whats special about the basis e_i ???

- Application-specific basis
- Fewer basis $\rightarrow$ faster computations

Optimality: What is the best n-dimensional space X_n for this representation? Then we will choose our basis as $e_i \in X_n$

Best approximation space X_n

$$h_\mu(f) \approx \sum_{i=1}^{n} c_i(\mu) e_i(f) \in X_n$$

Kolmogorov n-width of $\mathcal{F}$

$$d_n(\mathcal{F}) := \min_{\dim X_n \leq n} \; \max_{h_\mu \in \mathcal{F}} \left\| h_\mu - \sum_{i=1}^{n} c_i(\mu) e_i(f) \right\|$$

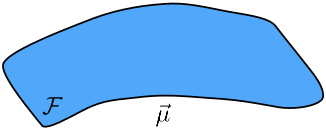
measures error of the best n-dimensional space X_n approximating $\mathcal{F}$

Bottleneck: Solving the n-width problem for X_n is in general not possible.

Practical implementation: Greedy method

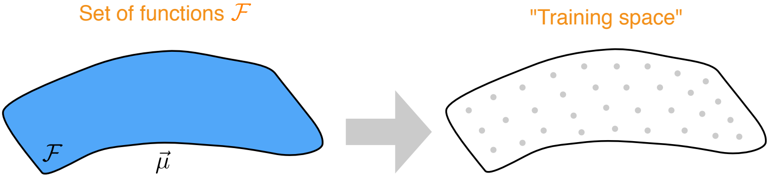
Can find an approximation space X_n that nearly satisfies the *n*-width

Set of functions $\mathcal{F}$



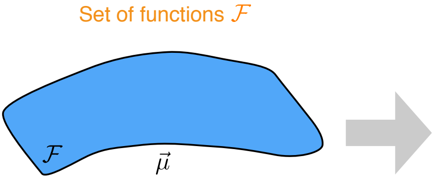
Practical implementation: Greedy method

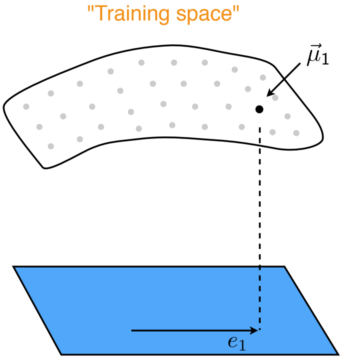
Can find an approximation space X_n that nearly satisfies the *n*-width



Practical implementation: Greedy method

Can find an approximation space X_n that nearly satisfies the *n*-width



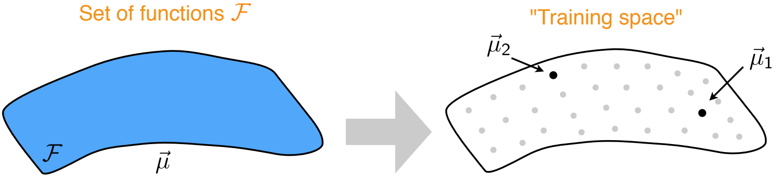


1) Choose any parameter,

$$e_1 = h(\vec{\mu_1}),\ C_1 = \{e_1\}$$

Practical implementation: Greedy method

Can find an approximation space X_n that nearly satisfies the *n*-width

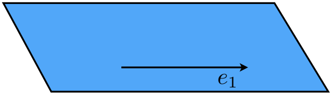


1) Choose any parameter,

$$e_1 = h(\vec{\mu_1}), \; C_1 = \{e_1\}$$

2) Greedy search - Find the parameter that maximizes:

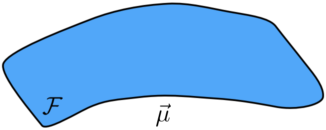
$$||h_{\vec{\mu}} - P_1(h_{\vec{\mu}})||, \; P_1(h_{\vec{\mu}}) = e_1 \langle e_1, h_{\vec{\mu}} \rangle$$



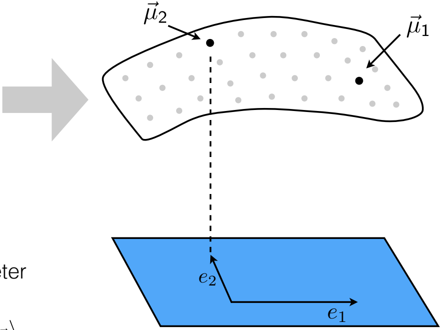
Practical implementation: Greedy method

Can find an approximation space X_n that nearly satisfies the *n*-width

Set of functions $\mathcal{F}$



"Training space"



1) Choose any parameter,

$$e_1 = h(\vec{\mu_1}),\ C_1 = \{e_1\}$$

2) Greedy search - Find the parameter that maximizes:

$$||h_{\vec{\mu}} - P_1(h_{\vec{\mu}})||,\ P_1(h_{\vec{\mu}}) = e_1\langle e_1, h_{\vec{\mu}}\rangle$$

3) Orthogonalization to get basis vector e_2

$$C_2 = \{e_1, e_2\},\ C_1 \subset C_2$$

Theorem (Binev+ 2011, DeVore+ 2012)

If the Kolmogorov n-width decays exponentially (or with polynomial order) so does the greedy approximation error $\sigma_n(\mathcal{F})$

$$d_n(\mathcal{F}) \leq Ce^{-c_0 n^\alpha} \quad \rightarrow \quad \sigma_n(\mathcal{F}) \leq \sqrt{2C}e^{-c_1 n^\alpha}$$

where C, c_0, α, *and* $c_1 := 2^{-1-2\alpha}c_0$ *are positive constants.*

Remarks

- X_n found through greedy algorithm *nearly optimal* compared to best space
- If we define an N-by-K matrix $A = [h_{\mu_1}(\mathbf{f}), \ldots, h_{\mu_K}(\mathbf{f})]$ the greedy selects n columns from A which serve as a low-rank approximation

Example basis generation

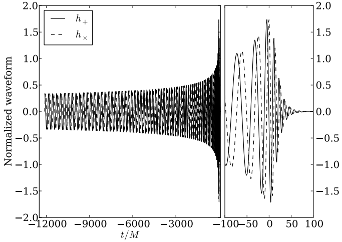
Effective one body (Pan et al., 2011)

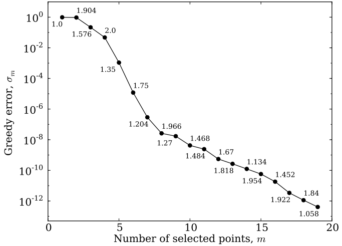
- (2,2) mode for $q \in [1, 2]$, duration $\approx 12,000M$
- Fast decay of approximation (overlap) error

$$\max_q \| h_q - \sum_{i=1}^{m} c_i(q) e_i \|^2$$

Other evidence

- Observed across models, regimes
- Observed by groups using POD/SVD
  - Cannon et al (PRD 044025)
  - M. Pürrer (arXiv:1402.4146)

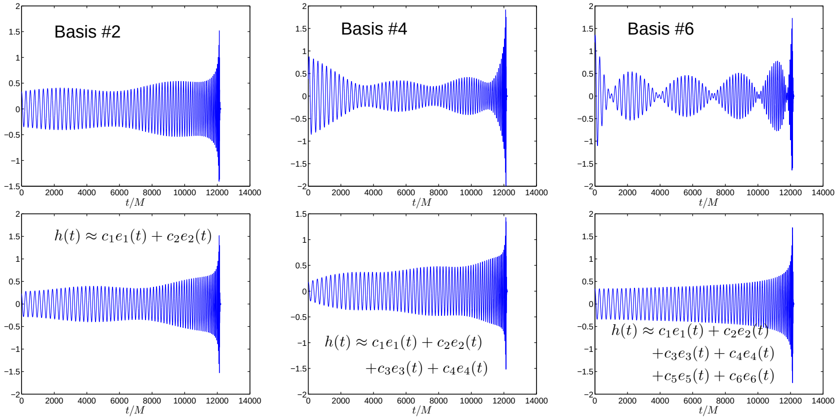




Waveform compression application (ex: $q \sim 1.2040$)

Ortho. Basis

Approx:



(a) 2 term, err ~ 1 (b) 4 term, err $\sim 10^{-1}$ (c) 6 term, err $\sim 10^{-6}$

Summary of step 1

We use a greedy algorithm to find a nearly optimal n-dimensional space X_n, whose basis are $e_i \in X_n$. We can represent gravitational waves as

$$h_\mu(f) \approx \sum_{i=1}^{n} c_i(\mu) e_i(f)$$

and the aproximation error is

$$\left| h_\mu(f) - \sum_{i=1}^{n} c_i(\mu) e_i(f) \right| \leq \sigma_n$$

where σ_n is computable, and for smooth models $\sigma_n \propto \mathrm{e}^{-n}$. On to step 2.

Where are the good points for integrating in X_n?

- Find interpolation nodes, derive an interpolatory quadrature rule
- In data analysis applications points *cannot* be freely drawn from Ω
- Naively selected points do not guarantee
  1. The interpolation problem is well-conditioned or even has a solution
  2. The interpolation error is small

Empirical interpolation method[1]

- Input: n basis $\{e_i(f)\}_{i=1}^{n}$
- Output: Nearly optimal selection of n times $\{F_i\}_{i=1}^{n}$
- These times are adapted to the problem/basis - unlike Chebyshev nodes
- Sequential selection of points: $\{F_1\} \to \{F_1, F_2\} \to \dots$

Algorithm

- Set of points $\{F_j\}_{j=1}^{i-1}$ for interpolation with the first $i-1$ basis
- To find F_i

$$F_i = \mathrm{argmax}_f \left| e_i(f) - \sum_{j=1}^{i-1} e_i(F_j)\hat{e}_j(f) \right|$$

[1] Barrault 2004, Maday 2009, Chaturantabut 2009, Sorensen 2009

Example: Points for polynomial interpolation

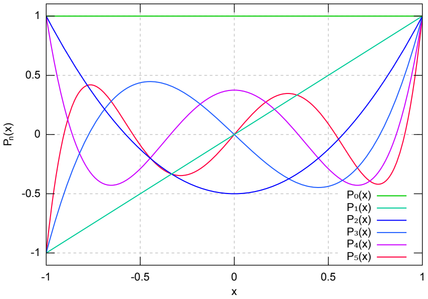
Basis are normalized polynomials defined on $[-1, 1]$

$$P_0(x) = \frac{1}{\sqrt{2}}$$

$$P_1(x) = \sqrt{\frac{3}{2}}x$$

$$P_2(x) = \sqrt{\frac{5}{8}}\left(3x^2 - 1\right)$$

$$\vdots$$



Q: Where are the “good” interpolation points?

Example: Points for polynomial interpolation

Basis:

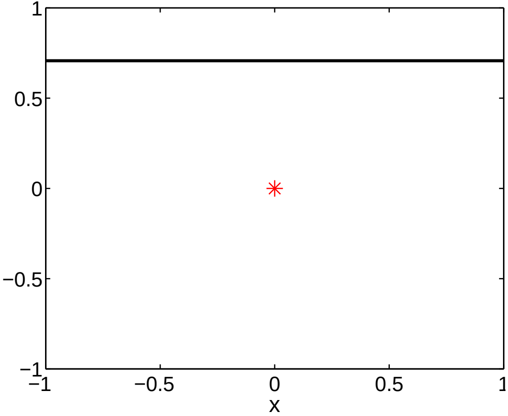
$$P_0(x) = \frac{1}{\sqrt{2}}$$

Residual:

$$P_0(x) - 0 = \frac{1}{\sqrt{2}}$$

Point selection (no preference):

$$x = 0$$



Example: Points for polynomial interpolation

Basis:

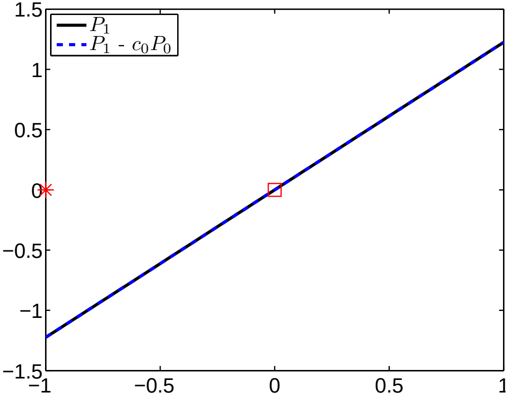
$$P_1(x) = \sqrt{\frac{3}{2}}x$$

Residual:

$$P_1(x) - c_0 P_0 = \sqrt{\frac{3}{2}}x$$

Point selection (either ± 1):

$$x = -1$$



Example: Points for polynomial interpolation

Basis:

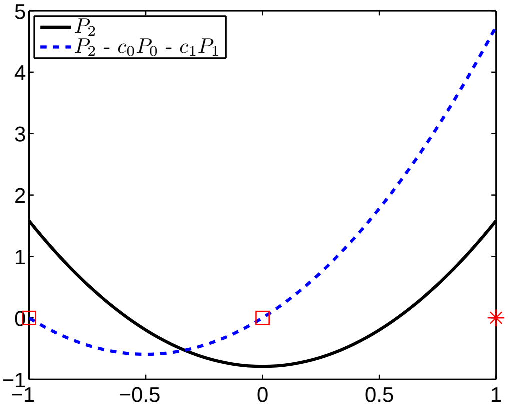
$$P_2(x) = \sqrt{\frac{5}{8}}\,(3x^2 - 1)$$

Residual:

$$P_2(x) - (c_0 P_0 + c_1 P_1)$$

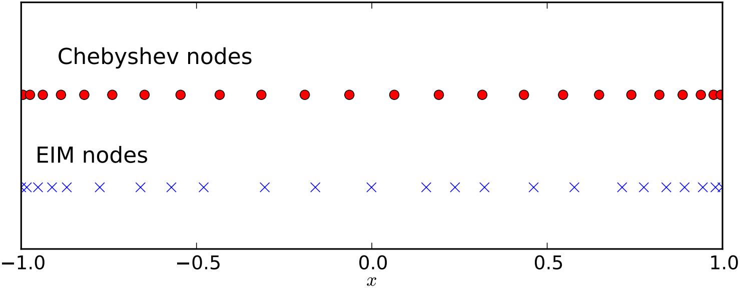
Point selection:

$$x = 1$$



Example: Points for polynomial interpolation/integration

Continue the process until # points = # basis. First 24 points for 24 polynomial basis...



Distribution and approximation error properties similar to Chebyshev nodes

Interpolant = basis + points

Using the n points $\{F_i\}_{i=1}^n$ and basis $\{e_i\}_{i=1}^n$, any h_μ can be written as

$$\boxed{\mathcal{I}_n[h_\mu](f) = \sum_{i=1}^{n} h_\mu(F_k)\hat{e}_i(f)}$$

where $n \ll N$ (N= data's length)

The ROQ approximation

Empirical interpolant

The ROQ rule is completed as follows:

$$\langle d, h_\mu \rangle = \Delta f \sum_{i=1}^{N} \frac{d^*(f_i) h_\mu(f_i)}{\sigma_i^2(f_i)} \approx \Delta f \sum_{i=1}^{N} \frac{d^*(f_i) \mathcal{I}_n[h_\mu](f_i)}{\sigma_i^2(f_i)} = \sum_{i=1}^{n} \omega_i h_\mu(F_i) = \langle d, h_\mu \rangle_{\texttt{ROQ}}$$

where the data-specific weights ω comprise a startup cost.

The ROQ approximation

Empirical interpolant

The ROQ rule is completed as follows:

$$\langle d, h_\mu \rangle = \Delta f \sum_{i=1}^{N} \frac{d^*(f_i) h_\mu(f_i)}{\sigma_i^2(f_i)} \approx \Delta f \sum_{i=1}^{N} \frac{d^*(f_i) \mathcal{I}_n[h_\mu](f_i)}{\sigma_i^2(f_i)} = \sum_{i=1}^{n} \omega_i h_\mu(F_i) = \langle d, h_\mu \rangle_{\texttt{ROQ}}$$

where the data-specific weights ω comprise a startup cost.

Error bounds [SF+, J. of Scientific Computing]

Given the greedy approximation error $\sigma_n(\mathcal{F})$ and $\Lambda_n = \|\mathcal{I}_n\|_2$

$$\left| \langle d, h_\mu \rangle - \langle d, h_\mu \rangle_{\texttt{ROQ}} \right| \; < \; \sigma_n(\mathcal{F}) \Lambda_n \|d\| \, \|h_\mu\|$$

For smooth GW models, convergence exponentially fast with n

Outline

Offline: Decide on...

- GW model,
- Detector settings (sampling rate, flow, fmax),
- Parameter domain.

Compute basis + integration nodes and save to file

Offline: Decide on...

- GW model,
- Detector settings (sampling rate, flow, fmax),
- Parameter domain.

Compute basis + integration nodes and save to file

Online: Assemble data-specific reduced-order quadrature rule fast likelihood evaluations. This requires the computation of the ROQ weights.

ROQ catalog

- Models
  - IMRPhenomPv2 (Smith+ PRD 94 2016)
  - IMRPhenomPv2 with non-GR deviations (Meidam+ PRD 2018)
  - IMRPhenomPv2_NRTidalv2 (unpublished)
  - SEOBNRv2_ROM_DoubleSpin (unpublished)
  - LackeyTidal2013_SEOBNRv2_ROM (unpublished)
  - SEOBNRv4T_surrogate (unpublished)
- Detectors
  - Current ground-based detectors LIGO, Virgo, KAGRA
  - Preliminary look at BNS signals with future detectors CE and ET (Smith+ PRL 2021)
- Future/ongoing work
  - IMRPhenomXPHM, beyond-GR extensions of the Phenom families, SEOBNRv4HM_ROM SEOBNRv4HM_NRTidalv2_NSBHv2
  - Carl-Johan Haster, Michael Pürrer, Rory Smith, and others

Using ROQs

- LALInference
- Bilby
- Parallel Bilby
- Example ROQ dataset (basis + nodes): `https://git.ligo.org/lscsoft/ROQ_data`

Building ROQs

Greedy and EI methods have nice theoretical *and* computational properties[2]

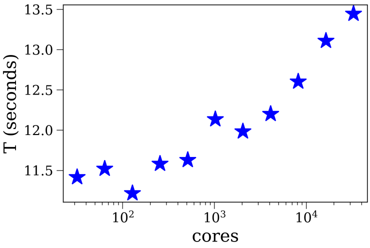


Figure: Time to complete 100 basis (n), for Training set $= 100 * \mathrm{cores}$. Cores are increased from 32 to $32,768$, with the largest matrix having $3,276,800$ columns. Machine: BlueWaters.

- Faster and parallelizes easier than, say, singular value decomposition
- Under-the-hood: Fast EIM algorithm (Field+ 2013) and iterative modified Gram-Schmidt (Hoffmann 1989)
- Automated ROQ building, validation, enrichment, and exports ROQ rule for LALInference and Bilby
- (Alternative) Python code (Qi and Raymond): `github.com/qihongcat/PyROQ`
- (Alternative) LALInferenceGenerateROQ

[2] https://bitbucket.org/sfield83/greedycpp/

Example with PhenomPv2 (Smith et al PRD 2016)

PhenomPv2 (Hannam, Schmidt, et al. PRL 2014):

- An IMR signal of precessing binary black holes
- Models precession by rotating the waveforms of an aligned-spin model PhenomD ("twist up" approach)
- Includes mode content for $(2, \{\pm 2, \pm 1, 0\})$

Case	Build strategy	f (Hz) Min	f (Hz) Max	Waveform duration T	Δf (Hz)	$\mathcal{M}$ ($M_\odot$) Min	$\mathcal{M}$ ($M_\odot$) Max
A	Enriched greedy	20	1024	$1.5\mathrm{s} \leq T \leq 4\mathrm{s}$	1/4	12.3	23
B	Enriched greedy	20	1024	$3\mathrm{s} \leq T \leq 8\mathrm{s}$	1/8	7.9	14.8
C	Enriched greedy	20	2048	$6\mathrm{s} \leq T \leq 16\mathrm{s}$	1/16	5.2	9.5
D	Enriched greedy	20	2048	$12\mathrm{s} \leq T \leq 32\mathrm{s}$	1/32	3.4	6.2
E	Enriched greedy	20	2048	$23.8\mathrm{s} \leq T \leq 64\mathrm{s}$	1/64	2.2	4.2
F	Enriched greedy	20	4096	$47.5\mathrm{s} \leq T \leq 128\mathrm{s}$	1/128	1.4	2.6

Offline (data independent): basis and ROQ points

Decide on a suitable range of parameter values, run greedy and EIM

- Sample in chirp mass, mass ratio, and spin-related parameters
- Training set size: $64^2 \times 8^5 = 134,217,728$ waveforms (terabytes in memory)
- Used XSEDE supercomputers and parallelized greedycpp code

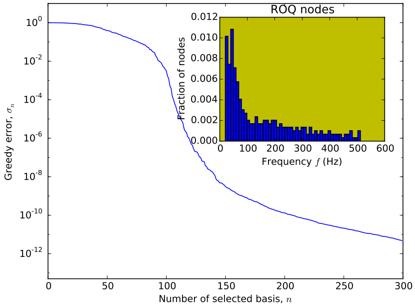


Figure: Greedy error and empirical interpolant (ROQ) nodes

Offline (data independent): validation

- Out-of-sample validation of the basis and empirical interpolant (basis + nodes)
- ROQ error is essentially empirical interpolant's error

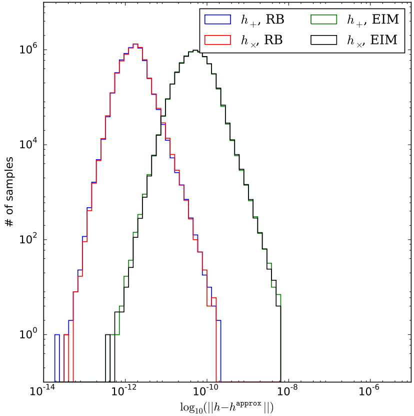
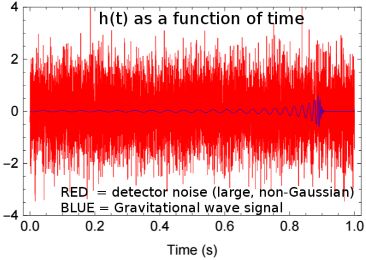


Figure: Out-of-sample errors

Startup: a signal has been detected!



Compute the data-dependent weights:

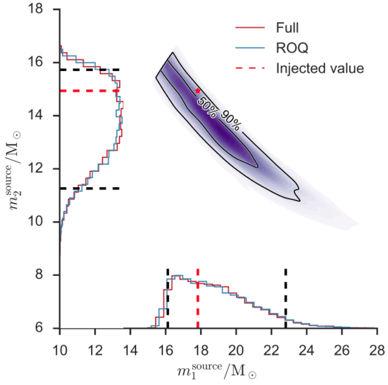
$$\vec{\omega}^{\,T} = \vec{E}^{\,T} V^{-1}$$

$$E_j = \Delta f \sum_{i=1}^{N} \frac{d^*(f_i) e_j(f_i)}{\sigma_i^2}$$

where V is the interpolation matrix.

Sample distribution $p(\mu|s)$, where the likelihood $P(s|\mu)$ uses a standard or ROQ

	m_1	m_2	d	$\lvert S_1 \rvert$	$\lvert S_2 \rvert$
ROQ	$28.72^{29.90}_{25.74}$	$21.50^{24.16}_{19.45}$	$269.5^{493.3}_{165.9}$	$0.6960^{0.8771}_{0.1643}$	$0.4466^{0.841}_{0.05357}$
Full	$28.72^{29.90}_{25.74}$	$21.50^{24.16}_{19.45}$	$269.5^{493.3}_{165.9}$	$0.6960^{0.8771}_{0.1643}$	$0.4466^{0.841}_{0.05357}$



Consistent values are a code sanity test: Due to error estimates we are guaranteed accuracy.

How much faster?

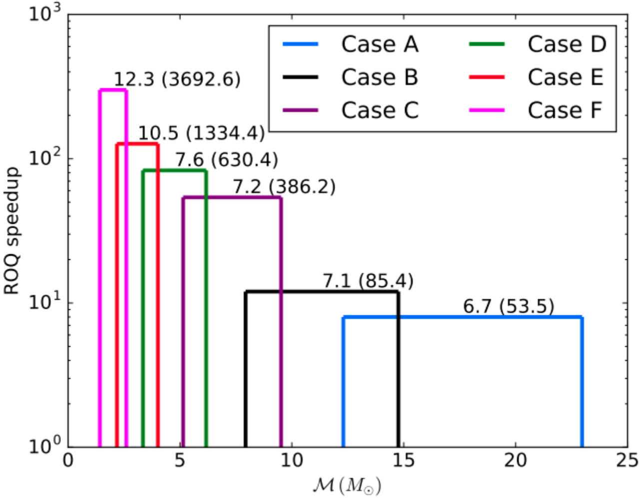


Figure: Annotated with the time (in hours) to compute 2×10^7 ROQ (Full) likelihood evaluations

Accelerating tests of GR

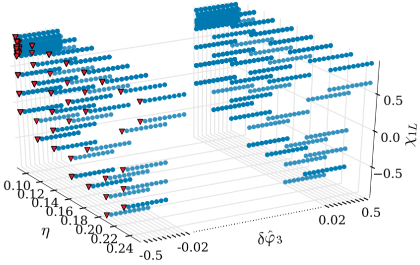


Figure: ROQ training set. Original basis for IMRPhenomPv2 (triangles) is extended in the additional parameter dimension for non-GR deviations. (Meidam, et al. PRD 2018)

- ROQs for non-GR deviation of IMRPhenomPv2
- A different ROQ for each of the 15 GR testing parameters
- Inference of testing parameters with and without ROQs are consistent
- Speedup factors from 3 (high mass) to 130 (low mass)

BNS events with third-generation observatories

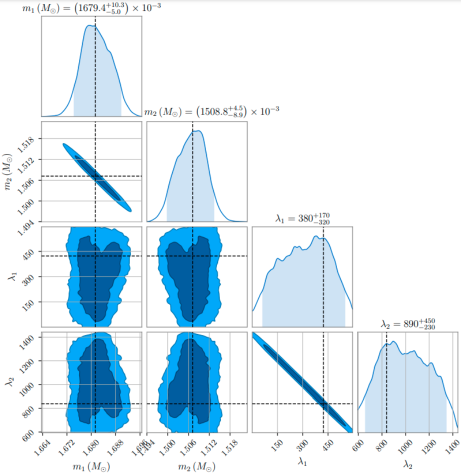


Figure: Posteriors for component masses and component tidal deformabilities computed with ROQ (Smith, et al. PRL 2021)

- Cosmic Explorer and Einstein Telescope
- IMRPhenomPv2_NRTidalv2
- BNS event at SNR = 2400
- fmin = 5Hz, fmax = 2048 Hz, duration = 90 min
- Takes 10 hours on 10 nodes, 16-cores/node (uses parallel Bilby and dynesty nested sampler)
- Without ROQ, would take (10 hours)(10^4)≈ 11 years

Summary

- Fast inference essential for keeping pace with detectors
- A model and data specific quadrature rule was developed
  - Works out-of-the-box for higher harmonic modes, complicated models, unknown "best fit parameters"
- Error bounds are rigorous and computable
- Production codes like greedycpp for ROQ building
- Accelerated parameter estimation studies by factors of 2 to 10^4
  - For 3G detectors, hours/days vs tens/hundreds years (estimated).
- Available in production codes LALInference, Bilby, parallel Bilby

Outlook/limitations/wish list

- Automate building ROQs for newest models? (Key practical limitation)
- Can more compact quadrature rules be found?
- Can similar strategies be applied to work with time-domain models?

Experiment 1: Comparison to Gaussian quadrature

Continuum

- $x \in [-1, 1]$ and weight $W(x) = 1$

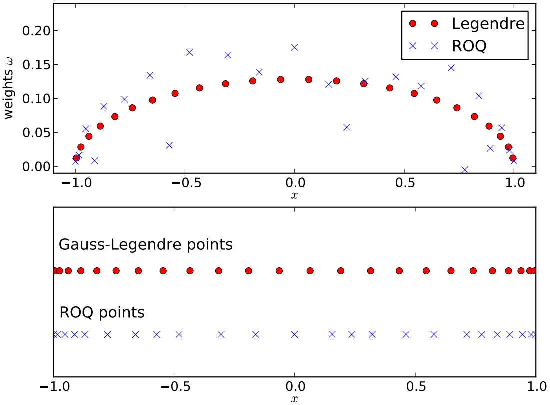
Discrete quadrature

- 24-point Gaussian quadrature

Reduced order quadrature

- 24 ROQ basis: Legendre polynomials, no greedy algorithm used
- 24 ROQ points: Subset of 1000 equidistant points sampling the basis

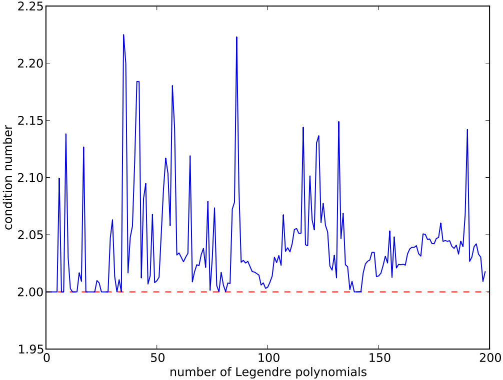
Point and weight distribution



Top: Weight ω_k and node $\{x_i\}$ distributions for each 24-point rule
Bottom: Quadrature node locations only

Conditioning of quadrature

- Negative weights can lead to poorly conditioned quadrature
- n-point ROQ rule for $n \in [2, 200]$



Condition number $\sum_{k=1}^{n} |\omega_k|$ for ROQ (blue) and GQ (red) rules