Application-specific quadrature for fast evaluation of parameterized inner products with noisy data

Scott Field

Department of Physics, University of Maryland

University of New Mexico, June 5, 2013

Joint work with Harbir Antil (George Mason), Priscilla Canizares (Cambridge), Frank Herrmann (UMD), Jonathan Gair (Cambridge), Ricardo Nochetto, (UMD), Manuel Tiglio (UMD).

Outline

Introduction

Reduced order quadratures

Experiments and applications
Consider the problem of integrating a function in 1 spatial dimension

\[ \int_{\Omega} f(x)W(x)dx \approx \sum_{i=1}^{N} f(x_i)\omega_i \]

Finding quadrature points \( x_i \) and weights \( \omega_i \) is well-studied

- Is \( f \) smooth? Use Gaussian quadratures for a standard \( W \)
- Is \( f \) non-smooth? Use trapezoidal or Simpson’s rule
- Error estimator? Gauss-Kronrod rule
Consider the parameterized problem in 1 spatial dimension

\[
\langle f, g \rangle (\mu, \nu) = \int_{\Omega} f^*(x; \mu) g(x; \nu) W(x) \, dx \approx \sum_{i=1}^{N} f^*(x_i; \mu) g(x_i; \nu) \omega_i
\]

computed with any ordinary quadrature rule with an integrand \( f^*(x)g(x) \)

Outlook

- If \( 10^6 \) values of \((\mu, \nu)\) are needed, each \( \approx 1s \), our code takes 12 days!
- We might design a custom quadrature rule tailored to our functions
- Invest time to build worthwhile if it's faster to use (and reuse)
Difficulties with parameterized integration

\[ \langle f, g \rangle (\mu, \nu) = \int_{\Omega} f^*(x; \mu)g(x; \nu)W(x)dx \]

Existing numerical quadrature rules could be expensive whenever...

- \( f(x; \mu) \) or \( g(x; \nu) \) are not well approximated by standard functions
- \( f(x; \mu) \) or \( g(x; \nu) \) highly oscillatory or different length scales
- \( f(x; \mu) \) is a stream of noisy data \( s(x) \), sampling dictated by experiment
- \( W(x) \) is something strange, perhaps empirically derived
Observations and strategies

\[ \langle f, g \rangle (\mu, \nu) = \int_{\Omega} f^*(x; \mu)g(x; \nu)W(x)dx \]

Some common situations...

- Needs to be computed for many values of \((\mu, \nu)\)
- Won’t know ahead of time which parameters to compute for
- Could be a serial procedure: selected \((\mu_i, \nu_i)\) depends on previous \(i - 1\)
- If \(g(x; \nu) = s(x)\) noisy data, integration often depends smoothly on \(\mu\)
Observations and strategies

\[ \langle f, g \rangle(\mu, \nu) = \int_{\Omega} f^*(x; \mu)g(x; \nu)W(x)dx \]

Plan of attack...

- Invest effort to build an application-specific quadrature rule \textit{offline}
- Once built it is reused \textit{online}, for example when new data is available
- If \( \langle f, g \rangle \) has smooth parametric dependence we expect \textit{fast, accurate} rule
Motivations

Gravitational waves emitted from two orbiting black holes. These sources could be in our galaxy or another one far, far away.
Motivations
Parameterized integrations in gravitational wave (GW) data analysis

1. A GW detector records some signal \( s(t) = h(t; \lambda) + n(t) \)
2. Noise \( |n(t)| \gg |h(\lambda)(t)| \)
3. Parameter estimation by correlating signal with model \( h(t; \mu) \) to recover parameter \( \lambda \)
4. Analysis can take hours to many months depending on data and model

1. Noise free signal \( h(t; \lambda) \)
Motivations
Parameterized integrations in gravitational wave (GW) data analysis

1. A GW detector records some signal \( s(t) = h(t; \lambda) + n(t) \)
2. Noise \(|n(t)| \gg |h_\lambda(t)|\)
3. Parameter estimation by correlating signal with model \( h(t; \mu) \) to recover parameter \( \lambda \)
4. Analysis can take hours to many months depending on data and model

2. Observed signal \( s(t) \)
Motivations
Parameterized integrations in gravitational wave (GW) data analysis

1. A GW detector records some signal \( s(t) = h(t; \lambda) + n(t) \)
2. Noise \( |n(t)| \gg |h_\lambda(t)| \)
3. Parameter estimation by correlating signal with model \( h(t; \mu) \) to recover parameter \( \lambda \)
4. Analysis can take hours to many months depending on data and model

3. To recover \( \lambda \) multiple evaluations of
\[
\int_{f_{\text{low}}}^{f_{\text{high}}} s^*(f)h(f; \mu)W(f)df
\]
and \( W(f) \) describes detector noise

Scott Field  ROQ for parameterized inner products with noisy data
Motivations
Parameterized integrations in gravitational wave (GW) data analysis

1. A GW detector records some signal $s(t) = h(t; \lambda) + n(t)$
2. Noise $|n(t)| \gg |h_\lambda(t)|$
3. Parameter estimation by correlating signal with model $h(t; \mu)$ to recover parameter $\lambda$
4. Analysis can take hours to many months depending on data and model

4. This may take a while
Preview of talk

- Algorithms to build application-specific quadrature rules for generic, parameterized integrals
- Work largely motivated by bottlenecks encountered in data analysis studies
- Examples typically draw from GW physics, however approach is general
Outline

Introduction

Reduced order quadratures

Experiments and applications
Problem Formulation

Parametrized Functions

Let
\[ \mathcal{F} := \{ h_\mu : \Omega \rightarrow \mathbb{C} \mid \mu \in \mathcal{P}, h_\mu \in \mathcal{C} \} \]
be a set of parametrized functions where $\Omega$, $\mathcal{P}$ denote the “physical” and parameter domains and $\mathcal{F}$ denotes a compact subset of a Hilbert space $\mathcal{H} \supset \mathcal{F}$.

- $h_\mu$ could be closed-form, solutions to ODEs or PDEs
- In data analysis context $h_\mu$ is the parameterized model

Inner Product Computation

Given two arbitrary parameters $\mu_1, \mu_2 \in \mathcal{P}$, consider
\[ \langle f, g \rangle (\mu_1, \mu_2) = \int_\Omega f_{\mu_1}^*(x) g_{\mu_2}(x) W(x) dx \]
Introduction to reduced order quadratures (ROQ)

ROQ roadmap

1. We have an existing quadrature rule and a set of functions $\mathcal{F}$
Introduction to reduced order quadratures (ROQ)

ROQ roadmap

1. We have an existing quadrature rule and a set of functions $\mathcal{F}$
2. Find an accurate and compact basis to represent any element of $\mathcal{F}$. The basis will be a non-standard, application-specific one
Introduction to reduced order quadratures (ROQ)

**ROQ roadmap**

1. We have an existing quadrature rule and a set of functions $\mathcal{F}$
2. Find an accurate and compact basis to represent any element of $\mathcal{F}$. The basis will be a non-standard, application-specific one
3. Find points in the physical domain $\Omega$ for good integration
   - Points *could* be a subset of the existing quadrature rule
   - Accurate and stable (recall Newton-Cotes becomes ill-conditioned)
Introduction to reduced order quadratures (ROQ)

ROQ roadmap

1. We have an existing quadrature rule and a set of functions $\mathcal{F}$
2. Find an accurate and compact basis to represent any element of $\mathcal{F}$. The basis will be a non-standard, application-specific one
3. Find points in the physical domain $\Omega$ for good integration
   - Points could be a subset of the existing quadrature rule
   - Accurate and stable (recall Newton-Cotes becomes ill-conditioned)
4. $\{x_i, \omega_i\}_{i=1}^N \rightarrow \{X_i, \omega_i^{\text{ROQ}}\}_{i=1}^n$. Typically $n \ll N$.

- Algorithms/framework draw from recent developments in model order reduction
Approximations

Approximation of parameterized functions $\mathcal{F}$ with an $n$-dimensional space $X_n$

$$\sup_{h_\mu \in \mathcal{F}} \inf_{f \in X_n} \| h_\mu - f \| \leq \epsilon$$

where $\epsilon$ is a user defined approximation tolerance ($\approx 10^{-6}$)

Non-adaptive approximations
- Space $X_n$ fixed and independent of $\mathcal{F}$
- Example: $X_n$ degree $n$ polynomials (Gaussian quadratures)

Adaptive approximations
- Space $X_n$ tailored to $\mathcal{F}$
- Example: Basis of $X_n$ drawn from $\mathcal{F}$ (reduced order quadratures)
When to seek adaptive approximations?

- Time invested to find adaptive approximations worthwhile
  - Expect to reuse information
- Non-adaptive approximations are poor
- High evaluation cost $h_{\mu}(x_i)$ at each $x_i \in \Omega$
  - Even moderately fewer $x_i$ will be useful

When will adaptive approximations converge quickly??
Kolmogorov $n$-width of $\mathcal{F}$ in $\mathcal{H}$

$$d_n(\mathcal{F}; \mathcal{H}) := \inf_{\dim X_n \leq n} \sup_{h_\mu \in \mathcal{F}} \inf_{f \in X_n} \| h_\mu - f \| = \inf_{\dim X_n \leq n} \sup_{h_\mu \in \mathcal{F}} \| h_\mu - P_n h_\mu \|,$$

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

Orthogonal projection $P_n : \mathcal{F} \to X_n$

$$h_\mu \approx P_n h_\mu := \sum_{i=1}^{n} \langle e_i, h_\mu \rangle e_i,$$

$P_n h_\mu$ is best representation of $h_\mu$ in $X_n$ and $\{e_i\}_{i=1}^{n}$ an orthonormal basis of $X_n$

Bottleneck: Sadly, finding $X_n$ is in general not possible!
Approximate solution to the $n$-width problem

1. Sample the continuum

Define \textit{training set} through sampling at parameter points $\mathcal{T}_K = \{\mu_i\}_{i=0}^K$.

$$\mathcal{F}_K = \{h_\mu \in \mathcal{F} : \mu \in \mathcal{T}_K\}$$

Note: Sampling must be dense enough
Approximate solution to the \( n \)-width problem

1. Sample the continuum
Define \textit{training set} through sampling at parameter points \( \mathcal{T}_K = \{\mu_i\}_{i=0}^K \)

\[
\mathcal{F}_K = \{ h_\mu \in \mathcal{F} : \mu \in \mathcal{T}_K \}
\]

Note: Sampling must be dense enough

2. Greedy strategy
Find \( F_n \approx \mathcal{F}_K \) by solving \( n \) easy problems
- Given \( F_i \) the algorithm optimally chooses \( F_{i+1} \) and continues to \( F_n \)
- Sequence of hierarchical spaces are constructed \( F_1 \subset F_2 \subset ... \subset F_n \)
Greedy algorithm (setup)

**Goal:** Find $F_n \approx \mathcal{F}$

1. Choose a parameter $\mathcal{P}$ and physical $\Omega$ domains
2. Sample continuum $\mathcal{P}$ with *dense* training set $\mathcal{T}_K = \{\mu_i\}_{i=0}^{K}$
3. Initialize algorithm with random $\mu_1$ and let $F_1 = \text{span}\{h_{\mu_1}\}$

To go from $F_i$ to $F_{i+1}$...
Greedy algorithm

Define greedy error \( \sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - \mathcal{P}_i h_\mu \| \)
Greedy algorithm

Define greedy error \( \sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - \mathcal{P}_i h_\mu \| \)

While \( \sigma_i(\mathcal{F}_K; \mathcal{H}) \geq \text{Tol} \)

\( i \rightarrow i + 1 \)
Greedy algorithm

Define greedy error

$$\sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - P_i h_\mu \|$$

While $$\sigma_i(\mathcal{F}_K; \mathcal{H}) \geq Tol$$

$$i \rightarrow i + 1$$

1. For all $$\mu \in \mathcal{T}_K$$ compute $$\| h_\mu - P_i h_\mu \|$$
Greedy algorithm

Define \textit{greedy error} \( \sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - P_i h_\mu \| \)

While \( \sigma_i(\mathcal{F}_K; \mathcal{H}) \geq \text{Tol} \)

\( i \rightarrow i + 1 \)

1. For all \( \mu \in \mathcal{T}_K \) compute \( \| h_\mu - P_i h_\mu \| \)
2. Find the parameter \( \mu_{i+1} \) which maximizes the error of step 1
Greedy algorithm

Define greedy error

\[ \sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - \mathcal{P}_i h_\mu \| \]

While \( \sigma_i(\mathcal{F}_K; \mathcal{H}) \geq \text{Tol} \)

\( i \rightarrow i + 1 \)

1. For all \( \mu \in \mathcal{T}_K \) compute \( \| h_\mu - \mathcal{P}_i h_\mu \| \)
2. Find the parameter \( \mu_{i+1} \) which maximizes the error of step 1
3. Let \( h_{i+1} = h_{\mu_{i+1}} \) and \( F_{i+1} = \text{span}\{h_1, \ldots, h_{i+1}\} \)
**Greedy algorithm**

Define *greedy error* \( \sigma_i(\mathcal{F}_K; \mathcal{H}) := \sup_{\mu \in \mathcal{T}_K} \| h_\mu - P_i h_\mu \| \)

While \( \sigma_i(\mathcal{F}_K; \mathcal{H}) \geq \text{Tol} \)

\[ i \to i + 1 \]

1. For all \( \mu \in \mathcal{T}_K \) compute \( \| h_\mu - P_i h_\mu \| \)
2. Find the parameter \( \mu_{i+1} \) which maximizes the error of step 1
3. Let \( h_{i+1} = h_{\mu_{i+1}} \) and \( F_{i+1} = \text{span}\{h_1, ..., h_{i+1}\} \)

**Output**: Collection of points \( \{\mu_i\}_{i=1}^n \) and corresponding basis \( \{h_i\}_{i=1}^n \)

**Result**: \( F_n = \text{span}\{h_i\}_{i=1}^n \) approximates training space \( \mathcal{F}_K \) up to \( \text{Tol} \)
**Result** [Binev 2011, DeVore 2012]: If \( n \)-width decays exponentially (or with polynomial order) so does the greedy error

\[
d_n(\mathcal{F}; \mathcal{H}) \leq C e^{-c_0 n^\alpha} \quad \rightarrow \quad \sigma_n(\mathcal{F}; \mathcal{H}) \leq \sqrt{2} C e^{-c_1 n^\alpha}
\]

where \( C, c_0, \alpha, \) and \( c_1 := 2^{1-2\alpha} c_0 \) are positive constants.
**Result** [Binev 2011, DeVore 2012]: If $n$-width decays exponentially (or with polynomial order) so does the greedy error

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

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

**Remarks**

- $F_N$ found through greedy algorithm nearly optimal compared to best space
Result [Binev 2011, DeVore 2012]: If $n$-width decays exponentially (or with polynomial order) so does the greedy error

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

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

Remarks

- $F_N$ found through greedy algorithm nearly optimal compared to best space
- If we define an $M$-by-$K$ matrix $A = [h_{\mu_1}(x), \ldots, h_{\mu_K}(x)]$ the greedy selects $n$ columns from $A$ which serve as a low-rank approximation.
**Result** [Binev 2011, DeVore 2012]: If $n$-width decays exponentially (or with polynomial order) so does the greedy error

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

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

**Remarks**

- $F_N$ found through greedy algorithm nearly optimal compared to best space
- If we define an $M$-by-$K$ matrix $A = [h_{\mu_1}(x), \ldots, h_{\mu_K}(x)]$ the greedy selects $n$ columns from $A$ which serve as a low-rank approximation
- Basis identified through greedy allows ROQ error to be controlled by $n$-widths thanks to Binev, DeVore, et al
Quadrature nodes

To complete the ROQ rule we must select nodes from physical domain $\Omega$

- What are good points for integrating in space $F_n$?
- In data analysis applications points cannot be freely drawn from $\Omega$
- Hierarchical nodal set advantageous
  - Faster to find
  - Leads to embedded ROQ rules

Preview: We will find $n$ nodes and derive an interpolatory quadrature formula
Recall a greedy algorithm has identified a basis \( \{ e_i \}_{i=1}^n \)

**Empirical interpolant**

- *If we know* \( n \) “good” nodes

\[
\{ X_i \}_{i=1}^n \subset \Omega
\]

*then* any \( h_\mu \in F \) can be written as

\[
I_n[h_\mu](x) := \sum_{i=1}^{n} c_i(\mu)e_i(x)
\]

where the \( c_i \) coefficients are solutions to the interpolation problem

\[
I_n[h_\mu](X_k) = h_\mu(X_k), \quad \forall \ k = 1, \ldots, n.
\]

- **ROQ rule is found by some version of** \( \int_\Omega I_n[h_\mu](x)dx \)
Empirical Interpolation Method\(^1\) (EIM)

- For application-specific bases where points are not known a-priori
- Algorithm selects interpolation points through a greedy criteria

Training set of physical points
Let \( \vec{x} = (x_1, x_2, \ldots, x_N)^T \) denote a vector of points where the set

\[
\{x_i\}_{i=1}^N \subset \Omega
\]

Goal: \( n \) points \( \{X_i\}_{i=1}^n \subset \{x\}_{i=1}^N \) such that

\[
\| h_{\mu} - I_n[h_{\mu}] \| \approx \sigma_n(\mathcal{F}; \mathcal{H})
\]

Recall best L2 approximation: \( \| h_{\mu} - P_n h_{\mu} \| \leq \sigma_n(\mathcal{F}; \mathcal{H}) \)

---

\(^1\)Barrault 2004, Maday 2009, Chaturantabut 2009, Sorensen 2009
Input: $n$ evaluated basis functions $\{\tilde{e}_i\}_{i=1}^n$, where $\tilde{e}_i = e_i(\bar{x})$
Input: $n$ evaluated basis functions $\{\tilde{e}_i\}_{i=1}^n$, where $\tilde{e}_i = e_i(\tilde{x})$

$i = \arg\max |\tilde{e}_1| \quad \textbf{Comment:} \arg\max$ returns the index of its largest entry.

Set $X_1 = x_i$
**Input:** \( n \) evaluated basis functions \( \{ \tilde{e}_i \}_{i=1}^{n} \), where \( \tilde{e}_i = e_i(\bar{x}) \)

\( i = \arg\max |\tilde{e}_1| \) \textbf{Comment:} \( \arg\max \) returns the index of its largest entry.

Set \( X_1 = x_i \)

For \( j = 2 \to n \) do
Input: \( n \) evaluated basis functions \( \{\tilde{e}_i\}_{i=1}^n \), where \( \tilde{e}_i = e_i(x) \)

\( i = \arg\max |\tilde{e}_1| \)  \textbf{Comment:} \( \arg\max \) returns the index of its largest entry. Set \( X_1 = x_i \)

For \( j = 2 \to n \) do

1. Find \( \mathcal{I}_{j-1}[e_j](\bar{x}) \)
Input: $n$ evaluated basis functions $\{\tilde{e}_i\}_{i=1}^n$, where $\tilde{e}_i = e_i(\vec{x})$

$i = \text{argmax } |\tilde{e}_1| \quad \text{Comment: } \text{argmax returns the index of its largest entry.}$
Set $X_1 = x_i$

For $j = 2 \rightarrow n$ do

1. Find $I_{j-1}[e_j](\vec{x})$
2. Compute the point-wise error $\vec{r} = I_{j-1}[e_j](\vec{x}) - \tilde{e}_j$
Input: *n evaluated* basis functions \( \{ \tilde{e}_i \}_{i=1}^n \), where \( \tilde{e}_i = e_i(\vec{x}) \)

\[
i = \text{argmax} |\tilde{e}_1| \quad \text{Comment: argmax returns the index of its largest entry.}
\]

Set \( X_1 = x_i \)

For \( j = 2 \rightarrow n \) do

1. Find \( I_{j-1}[e_j](\vec{x}) \)
2. Compute the point-wise error \( \vec{r} = I_{j-1}[e_j](\vec{x}) - \tilde{e}_j \)
3. \( i = \text{argmax} |\vec{r}| \)
Input: \( n \) evaluated basis functions \( \{\vec{e}_i\}_{i=1}^n \), where \( \vec{e}_i = e_i(\vec{x}) \)

\[ i = \arg\max |\vec{e}_1| \quad \textbf{Comment:} \ \arg\max \text{ returns the index of its largest entry.} \]

Set \( X_1 = x_i \)

For \( j = 2 \to n \) do

1. Find \( \mathcal{I}_{j-1}[e_j](\vec{x}) \)
2. Compute the point-wise error \( \vec{r} = \mathcal{I}_{j-1}[e_j](\vec{x}) - \vec{e}_j \)
3. \( i = \arg\max |\vec{r}| \)
4. Set \( X_j = x_i \)

Output: \( n \) points \( \{X_i\}_{i=1}^n \subset \{x_i\}_{i=1}^N \)
Interpolation Error Estimate

Let the set of greedy (reduced) basis \( \{ e_i \}_{i=1}^n \) be orthonormal and \( P_n h_\mu \in F_n \) be the optimal approximation of \( h_\mu \) with respect to the \( L^2 \)-norm. Then for every \( \mu \in \mathcal{P} \)

\[
\| h_\mu - I_n[h_\mu] \| \leq \Lambda_n \| h_\mu - P_n h_\mu \| \leq \Lambda_n \sigma_n(\mathcal{F}; \mathcal{H})
\]

where \( \Lambda_n = \| I_n \|_2 \) is a Lebesgue-like constant

- \( \Lambda_n \) is computable once basis and nodes are known
- No bounds on \( \Lambda_n \)'s growth with \( n \)
- Slow growth observed in practice
Standard quadrature

- Let \( \{\alpha_i, x_i\}_{i=1}^N \) denote quadrature weights and points then

\[
\int_\Omega h_\mu(x) dx \approx \sum_{i=1}^N \alpha_i h_\mu(x_i)
\]
Standard quadrature

- Let $\{\alpha_i, x_i\}_{i=1}^{N}$ denote quadrature weights and points then

$$\int_{\Omega} h_\mu(x) dx \approx \sum_{i=1}^{N} \alpha_i h_\mu(x_i)$$

Reduced order quadrature

- The set $\mathcal{F}$ is approximated by an $n$-dim space $F_n = \text{span}\{e_i\}_{i=1}^{n}$
- EIM points $\{X_i\}_{i=1}^{n}$ are accurate and well conditioned for interpolation in $F_n$

$$\sum_{i=1}^{N} \alpha_i h_\mu(x_i) \approx \sum_{i=1}^{N} \alpha_i I_n[h_\mu](x_i) = \sum_{i=1}^{n} \omega_i^{ROQ} h_\mu(X_i)$$

Numerical experiments show $n \ll N$
Define

\[ I_c = \int_{\Omega} h_\mu(x) \, dx, \quad I_d = \sum_{i=1}^{N} \alpha_i h_\mu(x_i), \quad I_{\text{ROQ}} = \sum_{i=1}^{n} \omega_{i}^{\text{ROQ}} h_\mu(X_i) \]

**ROQ error estimates**

Let \( \sigma_n(\mathcal{F}; \mathcal{H}) \leq \epsilon \) then \( \forall h_\mu \in \mathcal{F} \)

\[ |I_d - I_{\text{ROQ}}| < \sigma_n(\mathcal{F}; \mathcal{H})|\Omega|\Lambda_n \| h_\mu \|_d < \epsilon |\Omega|\Lambda_n \| h_\mu \|_d \]

where \( \sigma_n \) is the greedy error, \( \epsilon \) an error tolerance, and \( \Lambda_n = \| I_n \|_2 \)

\[ |I_c - I_{\text{ROQ}}| < |I_c - I_d| + \epsilon |\Omega|\Lambda_n \| h_\mu \|_d. \]

**Remarks**

- ROQ converges to \( I_d \) with same rate as \( n \)-width
- If \( I_d \approx I_c \) then convergence to exact result with same rate like \( n \)-width
Noisy data $s$

\[
\langle s, h_\mu \rangle \approx \sum_{i=1}^{N} \alpha_i s^*(x_i) h_\mu(x_i) \approx \sum_{i=1}^{N} \alpha_i s^*(x_i) \mathcal{I}_n h_\mu(x_i) = \sum_{i=1}^{n} \omega_i^{ROQ} h_\mu(X_i)
\]
Noisy data $s$

$$\langle s, h_\mu \rangle \approx \sum_{i=1}^{N} \alpha_i s^*(x_i) h_\mu(x_i) \approx \sum_{i=1}^{N} \alpha_i s^*(x_i) \mathcal{I}_n[h_\mu](x_i) = \sum_{i=1}^{n} \omega_i^{ROQ} h_\mu(X_i)$$

Parameterized products

$$\int_{\Omega} h^*_{\mu_i}(x) h_{\mu_j}(x) dx$$

- Approximation of $\widetilde{F} = \{ h^*_{\mu_i} h_{\mu_j} \mid h_{\mu_i}, h_{\mu_j} \in F \}$
- Two-step greedy leads to significantly faster offline building of basis
  - Training set for $\widetilde{F}$ uses greedy points found from $F_n \approx F$
A few considerations

Implementing the rule

- Finding basis and points could be costly – save output
- Someone gives you a good quadrature rule before deriving ROQ

Typical applications

- ROQ rule will be used over and over
  - Cost of building basis likely to outweigh single use
- You don’t know what parameters are ahead of time (e.g. data analysis)
- Naive quadrature has too many degrees of freedom (e.g. data analysis)
- Parameters drawn from continuum
  - If you know the parameters, store the results to file!
- Functions smooth – ROQ converges exponentially fast
Experiment setup

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 $\omega_k$ 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
Let $\mu_1, \mu_2 \in [-.1, .1]$ and consider integrals in 1 and 2 dimensions

$$\int_{-1}^{1} \left[(x - \mu_1)^2 + 0.1^2\right]^{-1/2} \quad \int_{-1}^{1} \int_{-1}^{1} \left[(x - \mu_1)^2 + (y - \mu_2)^2 + 0.1^2\right]^{-1/2}$$

- ROQ rule built from 150-point (for 1D) or $150^2$-point (for 2D) GQ rule.
- 2D GQ rule from tensor product grids
- ROQ nodal set formed by scattered point distributions tailored to the problem
Gravitational waves (GWs)

- Pair of orbiting black holes and/or neutron stars inspiral, merge, and ringdown
- Parameters of the binary system: objects’ masses (2 parameters), spins (6 parameters), and location/orientation in sky/detectors (8 parameters)

Courtesy: NASA GSFC
Gravitational wave detectors

- In absence of GWs the distance between two points is $L$
- A passing gravitational wave $h(t)$ causes small $\Delta L$ changes in length $L$.

Before GW passes by this ring of point masses has a radius $L$
Gravitational wave detectors

- In absence of GWs the distance between two points is $L$
- A passing gravitational wave $h(t)$ causes small $\Delta L$ changes in length $L$.

Single frequency, cross polarization

$$h(t) = h_x \sin(\omega t - kz)$$
Gravitational wave detectors

- In absence of GWs the distance between two points is $L$
- GW $h(t)$ causes small $\Delta L$ change in length – Expect $h(t) \propto \frac{\Delta L}{L} \leq 10^{-20}$

---

$^2$Fig. by Lee Lindblom
Gravitational wave detectors

- In absence of GWs the distance between two points is $L$
- GW $h(t)$ causes small $\Delta L$ change in length – Expect $h(t) \propto \frac{\Delta L}{L} \leq 10^{-20}$

\[ n(t) \]

\[ h(t), 100h(t) \]

\[ t (\text{sec.}) \]

\[ 0, 0.1, 0.2, 0.3, 0.4, 0.5 \]

\[ -2 \times 10^{-19}, -1 \times 10^{-19}, 0, 1 \times 10^{-19}, 2 \times 10^{-19} \]

\[ \times 10^{-19} \]

\[ 0, 0.1, 0.2, 0.3, 0.4, 0.5 \]

\[ 3 \]

\[ \text{Fig. by Lee Lindblom} \]
GW parameter estimation

- A detector alerts us to a signal in noisy data
- Correlate data with GW model to extract the physical parameters

Difficulties

- Model $h_\mu(t)$ described by high dimensional parameter space
- Data $s(t_i) = h_\lambda(t_i) + n(t_i)$ is a long time series, $\lambda$ true parameter
- $N$ equally spaced samples; $N = \text{(observation time)} \times \text{(sampling rate)}$
  - Ex: 32s at 4096Hz suggests $N \approx 130,000$ samples
- Cost to process data scales with $N$, dominated by evaluating model $h_\mu(t)$
GW Bayesian parameter estimation (I)

The (posterior) probability distribution function provides complete information about the parameters of the signal and is given by

\[ p(\mu|s) \propto P(s|\mu) \]

- \( p(\mu|s) \) is probability of parameters \( \mu \) given data \( s \)
- \( P(s|\mu) \) is the \textit{likelihood} that data \( s \) described by a particular \( \mu \)
- For Gaussian noise the likelihood is

\[
P(s|\mu) \propto \exp\left(-\chi^2/2\right), \quad \chi^2 = \langle s(f) - h_\mu(f), s(f) - h_\mu(f) \rangle
\]

which features Fourier transform of \( s(t) \) and \( h_\mu(t) \)
- Parameter estimation cost dominated by evaluation of \( \chi^2 \)
GW Bayesian parameter estimation (II)

Markov chain Monte Carlo (MCMC)
- We want to compute probability $p(\mu|s)$
- MCMC algorithms sample $p(\mu|s)$, efficient for high dimensional problems
- MCMC sequentially selects points, each requires evaluation of $\chi^2$
- Between hours and a year for algorithm to run!
Notice

\[ \chi^2 = \langle s, s \rangle + \langle h_\mu, h_\mu \rangle - 2\Re\langle s, h_\mu \rangle \]

- \( \langle s, s \rangle \) computed once
- \( \langle h_\mu, h_\mu \rangle \) has simple (often closed-form) expression

Standard computation

\[ \langle s, h_\mu \rangle \approx \Delta f \sum_{i=0}^{N} s(f_i)h^*_\mu(f_i) \]

where \( N \) is the number of data samples

- Widely (exclusively?) used for equally spaced, noisy data
- **Pros**: easy, robust. **Cons**: converges slowly with \( N \), expense of \( h_\mu(f_i) \)
- Model’s \( n \)-width (approximation properties) *independent* of data
Parameter estimation from “burst” signals

GW model

\[ h_\mu(t) = Ae^{-(t-t_c)^2/(2\alpha^2)} \sin(2\pi f_0(t - t_c)) , \]

describes merging black holes or supernovae GW signals.

- 4 dimensional model \( \mu = (A, t_c, \alpha, f_0) \)

Detector model

- Data segments of 32 second intervals
- Sampling rate of 64Hz such that observation every 1/64 seconds
- Frequency domain data samples \( (32 \times 64)/2 \)
- White noise (set weight \( W = 1 \))
  - Same average amplitude \( |n(f_i)| \) at each frequency component \( f_i \)
Offline (data independent)

Decide on suitable range of parameters, run greedy algorithm

Left: \((\alpha, f_0)\) points selected by the greedy algorithm.

Right: Approximation error \(\|h_\mu - P_n h_\mu\|_2^2\) as a function of greedy basis.
Offline (data independent)

Identify ROQ nodes from empirical interpolation method

Left: EIM points $\{F_i\}_{i=1}^{54}$ selected by the EIM algorithm.
Right: Empirical interpolant approximation error $\| h_\mu - I_n h_\mu \|^2$ and error bound.
Summary so far

✓ Greedy basis and ROQ points stored to file.
✓ Verified accuracy of basis and interpolation points.
✓ ROQ rule for this set of functions “Good for all time”

Some signal has been recorded!! Carry out parameter estimation...

True signal parameters
$\alpha = 1$, $f_0 = 0.25$, $t_c = 0.1$, $A$ unfixed

Modeled noise
At each frequency $n(f_i) = \mathcal{N}(0, \sigma^2)$

Mock data: Prepare data $s = h + n$, recover parameters with MCMC
Startup (data dependent)

Compute weights

\[ \tilde{\omega}^T = \tilde{E}^T A^{-1} \]

\[ E_j := \sum_{k=1}^{N} s^*(f_k) e_j(f_k) \Delta f \]

where the \( j^{th} \) column of the matrix \( A \) is basis \( e_j \) evaluated at ROQ nodes

**Figure:** Real (red squares) and imaginary (blue diamonds) ROQ weights
Sample distribution $p(\mu|s)$ where likelihood $P(s|\mu)$ uses standard or ROQ

$$\langle s, h_\mu \rangle = \Delta f \sum_{i=1}^{N} s^*(f_i) h_\mu(f_i) \approx \sum_{i=1}^{n} \omega_i h_\mu(F_i)$$

Left: Runtime. With $10^8$ points standard $\approx 1$ day, ROQ $\approx 1$ hour!!
Right: Speed-up of MCMC algorithm using a standard and ROQ quadrature

Scott Field
ROQ for parameterized inner products with noisy data
### Recovered values

<table>
<thead>
<tr>
<th>SNR Method</th>
<th>SNR</th>
<th>$f_0$</th>
<th>$\alpha$</th>
<th>$t_c$</th>
<th>$A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full</td>
<td>5</td>
<td>0.217 ± 0.069</td>
<td>0.896 ± 0.194</td>
<td>0.068 ± 0.104</td>
<td>1.704 ± 0.379</td>
</tr>
<tr>
<td>ROQ</td>
<td>5</td>
<td>0.217 ± 0.068</td>
<td>0.897 ± 0.196</td>
<td>0.069 ± 0.104</td>
<td>1.702 ± 0.375</td>
</tr>
<tr>
<td>Full</td>
<td>10</td>
<td>0.212 ± 0.048</td>
<td>0.875 ± 0.132</td>
<td>0.084 ± 0.053</td>
<td>2.362 ± 0.278</td>
</tr>
<tr>
<td>ROQ</td>
<td>10</td>
<td>0.209 ± 0.050</td>
<td>0.866 ± 0.132</td>
<td>0.085 ± 0.052</td>
<td>2.387 ± 0.287</td>
</tr>
<tr>
<td>Full</td>
<td>20</td>
<td>0.225 ± 0.029</td>
<td>0.891 ± 0.093</td>
<td>0.092 ± 0.028</td>
<td>2.944 ± 0.176</td>
</tr>
<tr>
<td>ROQ</td>
<td>20</td>
<td>0.224 ± 0.029</td>
<td>0.892 ± 0.093</td>
<td>0.093 ± 0.028</td>
<td>2.944 ± 0.177</td>
</tr>
<tr>
<td>Full</td>
<td>40</td>
<td>0.248 ± 0.009</td>
<td>0.981 ± 0.041</td>
<td>0.097 ± 0.016</td>
<td>3.471 ± 0.157</td>
</tr>
<tr>
<td>ROQ</td>
<td>40</td>
<td>0.248 ± 0.009</td>
<td>0.981 ± 0.042</td>
<td>0.097 ± 0.016</td>
<td>3.471 ± 0.157</td>
</tr>
</tbody>
</table>

\[
\mu_i = \frac{1}{N_{\text{mcmc}}} \sum_{j=1}^{N_{\text{mcmc}}} x_j^i \\
\sigma_i^2 = \frac{1}{N_{\text{mcmc}}} \sum_{j=1}^{N_{\text{mcmc}}} \left( x_j^i - \mu_i \right)^2
\]
Features

- Startup cost \( \approx \) time to compute inner products of data with basis (fast)
- Once weights specified, evaluations of \( \chi^2 \) about 25 times faster
- Accuracy in recovered parameters is preserved
What about more complicated GW signals?

GW signal from two orbiting black holes ("chirp" signal)
Two black holes of masses $m_1$ and $m_2$ rotate one another for long times

$$h_\mu(f) = Af^{-7/6} \cdot \exp \left( i \left\{ -\frac{\pi}{4} + \frac{3}{128} \left( \pi \cdot \frac{G}{c^3} \cdot f \cdot M_c \right)^{-5/3} \right\} + \ldots \right),$$

where $\mu = M_c = (m_1 m_2)^{3/5} (m_1 + m_2)^{-1/5}$.

$P = [A, B]$ where $A = 5 \times 10^{30}$ Kg and $B = 50 \times 10^{30}$ Kg
Detector’s noise curve

\[ S(y) = 9 \times 10^{-46} \left[ (4.49y)^{-56} + 0.16y^{-4.52} + 0.52 + 0.32 \cdot y^2 \right], \quad y = \frac{f}{150\text{Hz}} \]

is experimentally determined and implies a weight \( W = S^{-1} \)

Parameterized inner products

\[
\int_{40}^{360} h_{\mu_1}^*(f) h_{\mu_2}(f) W(f) df
\]

where \( \mu_1, \mu_2 \in \mathcal{P} \)

Building the ROQ

- Uses a two-step greedy approximate integrands \( h_{\mu_1}^*(f) h_{\mu_2}(f) W(f) \)
Inner product errors using i) Gauss-Legendre quadrature, ii) trapezoidal, iii) ROQ built from GQ, and iv) ROQ built from the trapezoidal

- Similar behavior between both ROQ rules (same basis)
- Only factor of 2 savings compared to GQ (predetermined points)
- Factor of 50 when using equally spaced “data” samples
Summary

- Introduced application/data specific quadrature for parameterized integrals
- Motivated by need to perform fast, accurate GW parameter estimation
- ROQ error decays like Kolmogorov $n$-width times a Lebesgue-like constant
- Offline costs high, online significantly faster

Future work and open questions

- Implementation within existing GW analysis pipelines underway
- Uses as application specific nested quadrature rule?
- Better criteria to choose ROQ basis and points?
- Uses outside of data analysis?