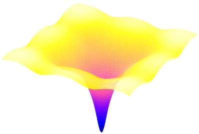
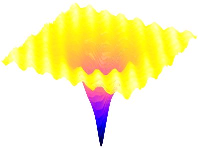
Software for Quasi-Monte Carlo and Fast Gaussian Process Regression

Aleksei G Sorokin[1,2] asorokin@hawk.iit.edu alegresor.github.io

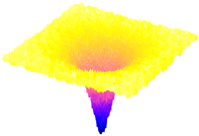
SE grid
L_2 relative error = 3.71e-2
time = 2.24e0
average CI width = 2.45e-3
CI captured 0.1%



SI lattice
L_2 relative error = 2.85e-2
time = 1.92e-3
average CI width = 3.71e0
CI captured 92.5%



DSI digital net
L_2 relative error = 2.62e-2
time = 4.54e-3
average CI width = 2.74e0
CI captured 98.8%



In collaboration with: Fred J Hickernell[1] (PhD Advisor), Sou-Cheng T Choi[1],
Pieterjan M Robbe[2], Jagadeeswaran Rathinavel[3]
Illinois Tech, Department of Applied Math[1]. Sandia National Lab[2]. Torc Robotics[3]

Quasi-Monte Carlo (QMC) Background

Efficient algorithms for high dimensional numerical integration

Quasi-Monte Carlo (QMC) Background

Efficient algorithms for high dimensional numerical integration

- QMC methods replace IID points with low-discrepancy (LD) points
- LD sequences evenly cover the unit cube in high dimensions
- QMC has faster convergence than IID Monte Carlo for nicely behaved functions [Dick and Pillichshammer, 2010, Dick et al., 2013, Niederreiter, 1992]
- Randomizing LD sequences improves convergence and enables error estimation [Dick, 2011, L'Ecuyer, 2016, L'Ecuyer et al., 2023, Owen, 1995, 2003]

Quasi-Monte Carlo (QMC) Background

Efficient algorithms for high dimensional numerical integration

- QMC methods replace IID points with low-discrepancy (LD) points
- LD sequences evenly cover the unit cube in high dimensions
- QMC has faster convergence than IID Monte Carlo for nicely behaved functions [Dick and Pillichshammer, 2010, Dick et al., 2013, Niederreiter, 1992]
- Randomizing LD sequences improves convergence and enables error estimation [Dick, 2011, L'Ecuyer, 2016, L'Ecuyer et al., 2023, Owen, 1995, 2003]

Applications

- **Financial modeling**, especially for option pricing [Giles and Waterhouse, 2009, Lai and Spanier, 1998, L'Ecuyer, 2009]
- **Solving PDEs**, often with random coefficients [Graham et al., 2011, 2015, Kuo and Nuyens, 2016, Kuo et al., 2012, 2015]
- **Ray tracing**, for simulating light transport in graphics rendering [Jensen et al., 2003, Raab et al., 2006, Waechter and Keller, 2011]

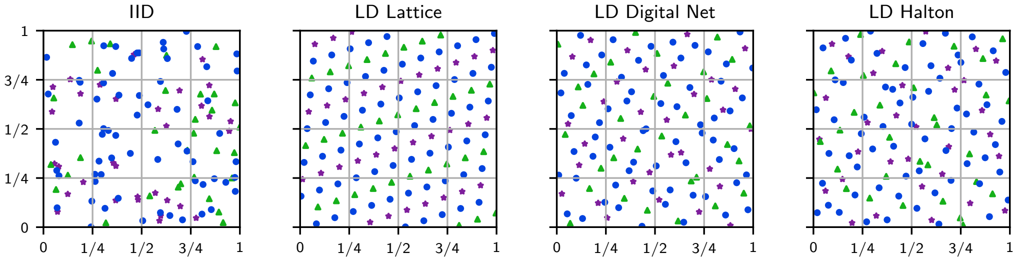
QMC Math

Efficient algorithms for high dimensional numerical integration

$$\mu = \int_{\mathcal{T}} g(\boldsymbol{t})\lambda(\mathrm{d}\boldsymbol{t}) = \int_{[0,1]^d} f(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \approx \frac{1}{n}\sum_{i=0}^{n-1} f(\boldsymbol{x}_i), \qquad \boldsymbol{x}_0,\dots,\boldsymbol{x}_{n-1} \in [0,1]^d$$

Classic Monte Carlo has error like $\mathcal{O}(1/\sqrt{n})$ using IID points $(\boldsymbol{x}_i)_{i=0}^{n-1}$

QMC has errors like $\mathcal{O}(1/n)$ using low discrepancy (LD) points with greater uniformity



QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

- **Randomized low-discrepancy sequences**
  - **Lattices:** random shifts
  - **Digital nets:** random digital shifts, LMS, Owen scrambling (NUS), higher-order nets
  - **Halton points:** digital shifts, permutation scrambles, NUS

QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

- **Randomized low-discrepancy sequences**
  - **Lattices:** random shifts
  - **Digital nets:** random digital shifts, LMS, Owen scrambling (NUS), higher-order nets
  - **Halton points:** digital shifts, permutation scrambles, NUS
- **Automatic transforms:** rewrite problem into $f:[0,1]^d \to \mathbb{R}$

QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

- **Randomized low-discrepancy sequences**
  - **Lattices:** random shifts
  - **Digital nets:** random digital shifts, LMS, Owen scrambling (NUS), higher-order nets
  - **Halton points:** digital shifts, permutation scrambles, NUS
- **Automatic transforms:** rewrite problem into $f:[0,1]^d \to \mathbb{R}$
- **Diverse use cases:** financial options, parameterized PDEs, sensitivity indices, ...

QMCPy Software

`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

- **Randomized low-discrepancy sequences**
  - **Lattices:** random shifts
  - **Digital nets:** random digital shifts, LMS, Owen scrambling (NUS), higher-order nets
  - **Halton points:** digital shifts, permutation scrambles, NUS
- **Automatic transforms:** rewrite problem into $f:[0,1]^d \to \mathbb{R}$
- **Diverse use cases:** financial options, parameterized PDEs, sensitivity indices, ...
- **Adaptive stopping criteria:** choosing n to meet user-specified error tolerance ε
  - **IID Monte Carlo** with guaranteed confidence intervals [Hickernell et al., 2013]
  - **QMC with replications**, see [L'Ecuyer et al., 2023] or [Owen, 2018, Chapter 17]
  - **QMC via decay tracking** of Fourier or Walsh coefficients [Hickernell et al., 2017]
  - **QMC via Bayesian cubature** using fast Gaussian processes [Rathinavel, 2019]
  - **Multilevel IID Monte Carlo** [Giles, 2008, Robbe et al., 2016, 2019]
  - **Multilevel QMC** [Giles and Waterhouse, 2009, Robbe et al., 2017]

QMCPy Software

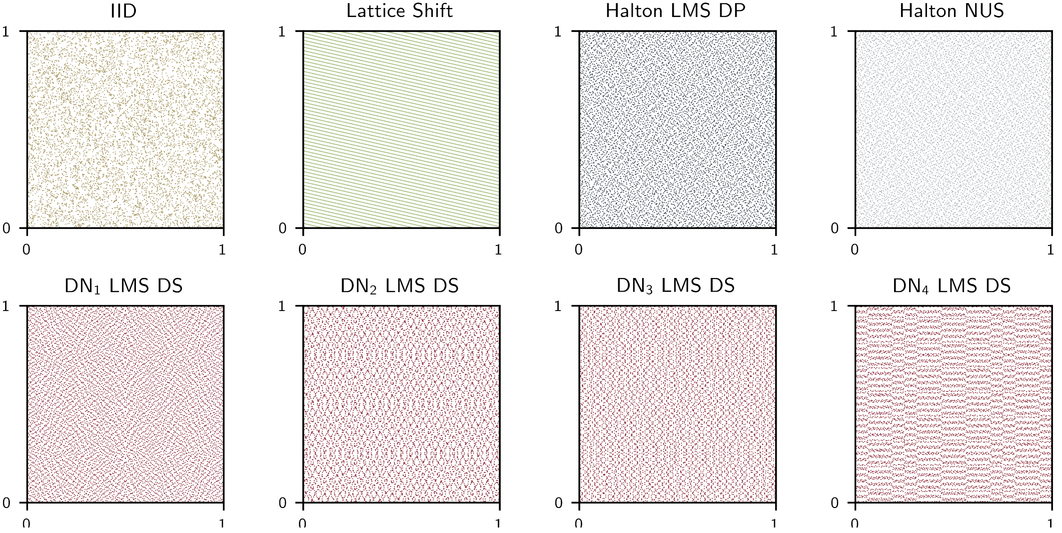
`pip install qmcpy` or visit qmcsoftware.github.io/QMCSoftware/

A Python software unifying QMC tools from across the literature [Choi et al., 2022a,b, Hickernell et al., 2025, Sorokin and Rathinavel, 2022]

- **Randomized low-discrepancy sequences**
  - **Lattices:** random shifts
  - **Digital nets:** random digital shifts, LMS, Owen scrambling (NUS), higher-order nets
  - **Halton points:** digital shifts, permutation scrambles, NUS
- **Automatic transforms:** rewrite problem into $f:[0,1]^d \to \mathbb{R}$
- **Diverse use cases:** financial options, parameterized PDEs, sensitivity indices, ...
- **Adaptive stopping criteria:** choosing n to meet user-specified error tolerance ε
  - **IID Monte Carlo** with guaranteed confidence intervals [Hickernell et al., 2013]
  - **QMC with replications**, see [L'Ecuyer et al., 2023] or [Owen, 2018, Chapter 17]
  - **QMC via decay tracking** of Fourier or Walsh coefficients [Hickernell et al., 2017]
  - **QMC via Bayesian cubature** using fast Gaussian processes [Rathinavel, 2019]
  - **Multilevel IID Monte Carlo** [Giles, 2008, Robbe et al., 2016, 2019]
  - **Multilevel QMC** [Giles and Waterhouse, 2009, Robbe et al., 2017]
- **Special kernels and fast transforms:** used for fast Gaussian process regression

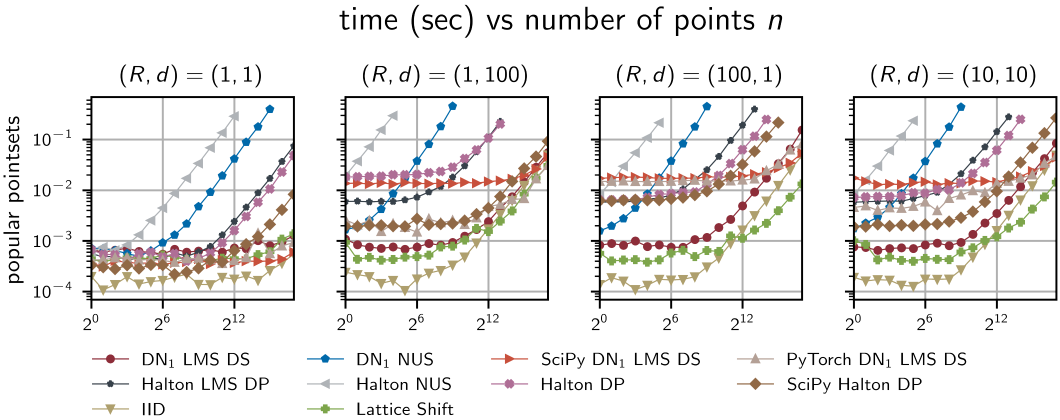
IID and Low-Discrepancy Points

Including higher-order digital nets and higher-order scramblings



Speed of Generating IID and Low-Discrepancy Points

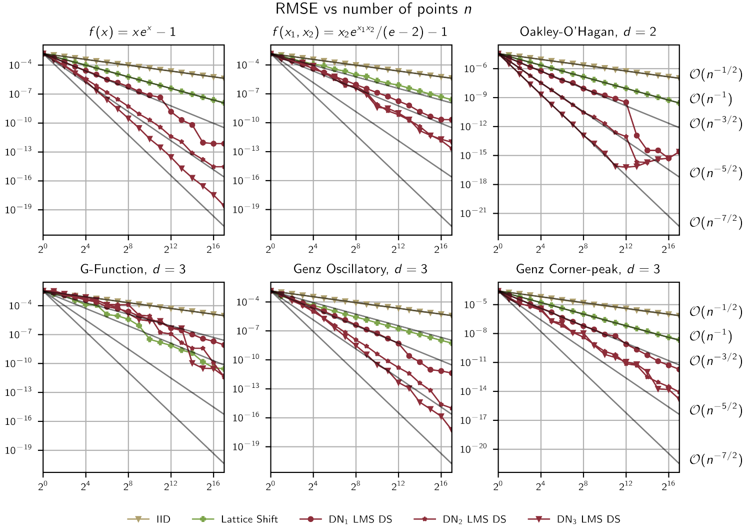
Randomized LD points are as fast to generate as IID points



R randomizations of n points in d dimensions

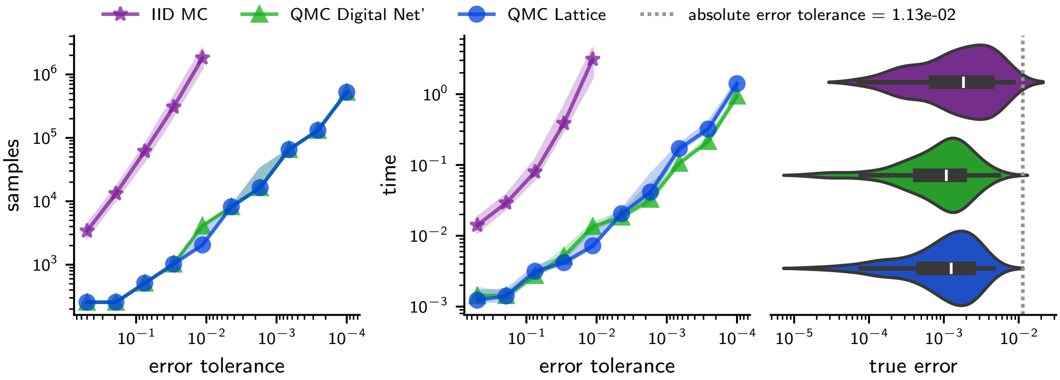
Root Mean Squared Error (RMSE) of Randomized QMC

Higher-order digital nets achieve higher-order convergence for low dimensional functions



Adaptive QMC for High Dimensional Asian Option Pricing

Automatically choose sample size n to meet user specified error tolerances



Gaussian Process (GP) Regression Background

Flexible interpolation models with built in UQ [Williams and Rasmussen, 2006]

Gaussian Process (GP) Regression Background

Flexible interpolation models with built in UQ [Williams and Rasmussen, 2006]

- UQ from (computable) posterior distribution conditioned on observations
- Equivalent to kernel interpolation in a RKHS (reproducing kernel Hilbert space)

Gaussian Process (GP) Regression Background

Flexible interpolation models with built in UQ [Williams and Rasmussen, 2006]

- UQ from (computable) posterior distribution conditioned on observations
- Equivalent to kernel interpolation in a RKHS (reproducing kernel Hilbert space)

Applications

- **Bayesian optimization**, for finding global minima of expensive simulations [Frazier, 2018, Snoek et al., 2012, Wu et al., 2020]
- **Solving PDEs**, with deterministic (1) or random (2) coefficients
  (1) [Batlle et al., 2025, Chen et al., 2021, 2024, Cockayne et al., 2017]
  (2) [Batlle et al., 2024, Kaarnioja et al., 2022, 2023, Sorokin et al., 2024]
- **Bayesian cubature**, possible since the integral of a GP is still Gaussian [Briol et al., 2019, O'Hagan, 1991, Rasmussen and Ghahramani, 2003]
- **Reliability analysis**, to efficiently predict the probability of system failure [Bae et al., 2020, Dubourg et al., 2013, Rackwitz, 2001, Renganathan et al., 2022, Sorokin and Rao, 2023, Zanette et al., 2018]

GPs Math

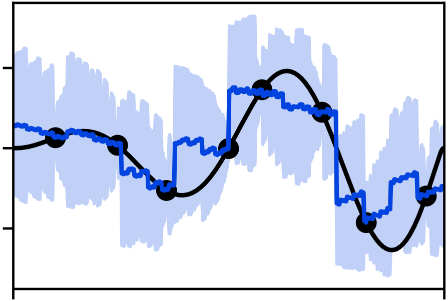
Flexible interpolation models with built in UQ [Williams and Rasmussen, 2006]

$f \sim \mathrm{GP}(0, K)$ with posterior mean and covariance

$$\mathbb{E}[f(\boldsymbol{x})|\mathsf{X}, \boldsymbol{f}] = \boldsymbol{K}_{\mathsf{X}}(\boldsymbol{x}) \mathsf{K}^{-1} \boldsymbol{f}$$

$$\mathbb{V}[f(\boldsymbol{x})|\mathsf{X}, \boldsymbol{f}] = K(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{K}_{\mathsf{X}}(\boldsymbol{x})^T \mathsf{K}^{-1} \boldsymbol{K}_{\mathsf{X}}(\boldsymbol{x})$$

- SPD $K : [0,1]^d \times [0,1]^d \to \mathbb{R}$
- Sampling locations $\mathsf{X} = \{\boldsymbol{x}_i\}_{i=0}^{N-1}$
- Sample values $\boldsymbol{f} = \{f(\boldsymbol{x}_i)\}_{i=0}^{N-1}$
- Kernel vector $\boldsymbol{K}_{\mathsf{X}}(\boldsymbol{x}) = \{K(\boldsymbol{x}, \boldsymbol{x}_i)\}_{i=0}^{N-1}$
- Gram matrix $\mathsf{K} = \{K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})\}_{i,i'=0}^{N-1}$



Typically requires $\mathcal{O}(N^2)$ storage and $\mathcal{O}(N^3)$ computations

Fast GPs Background

Reduce high GP costs by forcing structure into the Gram matrix [Zeng et al., 2006, 2009]

Fast GPs Background

Reduce high GP costs by forcing structure into the Gram matrix [Zeng et al., 2006, 2009]

- $\mathcal{O}(N \log N)$ computations (compared to typically $\mathcal{O}(N^3)$ cost), and
- $\mathcal{O}(N)$ memory (compare to classic $\mathcal{O}(N^2)$ memory requirements).

Fast GPs Background

Reduce high GP costs by forcing structure into the Gram matrix [Zeng et al., 2006, 2009]

- $\mathcal{O}(N \log N)$ computations (compared to typically $\mathcal{O}(N^3)$ cost), and
- $\mathcal{O}(N)$ memory (compare to classic $\mathcal{O}(N^2)$ memory requirements).

Fast GPs exploit Gram matrix structure present when pairing

- **Special low-discrepancy (LD) sequences**, either lattices or digital nets, to
- **(Digitally)-shift-invariant (SI or DSI) kernels**, of varying smoothness

Fast GPs Background

Reduce high GP costs by forcing structure into the Gram matrix [Zeng et al., 2006, 2009]

- $\mathcal{O}(N \log N)$ computations (compared to typically $\mathcal{O}(N^3)$ cost), and
- $\mathcal{O}(N)$ memory (compare to classic $\mathcal{O}(N^2)$ memory requirements).

Fast GPs exploit Gram matrix structure present when pairing

- **Special low-discrepancy (LD) sequences**, either lattices or digital nets, to
- **(Digitally)-shift-invariant (SI or DSI) kernels**, of varying smoothness

Requirements

- **Controlling design of experiments**, in order to use LD points
- **Using uncommon kernels**, either the SI or DSI kernels

Fast GPs Background

Reduce high GP costs by forcing structure into the Gram matrix [Zeng et al., 2006, 2009]

- $\mathcal{O}(N \log N)$ computations (compared to typically $\mathcal{O}(N^3)$ cost), and
- $\mathcal{O}(N)$ memory (compare to classic $\mathcal{O}(N^2)$ memory requirements).

Fast GPs exploit Gram matrix structure present when pairing

- **Special low-discrepancy (LD) sequences**, either lattices or digital nets, to
- **(Digitally)-shift-invariant (SI or DSI) kernels**, of varying smoothness

Requirements

- **Controlling design of experiments**, in order to use LD points
- **Using uncommon kernels**, either the SI or DSI kernels

Applications

- **Solving PDEs**, often with random coefficients [Kaarnioja et al., 2022, 2023, Sorokin et al., 2024]
- **Fast Bayesian cubature** [Rathinavel and Hickernell, 2019, 2022]
- **Discrepancy computation** [Hickernell, 1998a,b]

Fast GPs Pairing LD Points with Special Kernels

Fast GPs Pairing LD Points with Special Kernels

1. **LD Lattices + Shift Invariant (SI) Kernels**
   - Give circulant Gram matrices $\mathsf{K} = \{K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})\}_{i,i'=0}^{N-1}$

   $\therefore$ Eigendecomp $\mathsf{K} = \mathsf{V}\Lambda\overline{\mathsf{V}}$ where $\overline{\mathsf{V}}$ is the DFT matrix $\to$ FFT in $\mathcal{O}(N \log N)$

Fast GPs Pairing LD Points with Special Kernels

1. **LD Lattices + Shift Invariant (SI) Kernels**
   - Give circulant Gram matrices $\mathsf{K} = \{K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})\}_{i,i'=0}^{N-1}$
   - $\therefore$ Eigendecomp $\mathsf{K} = \mathsf{V}\Lambda\overline{\mathsf{V}}$ where $\overline{\mathsf{V}}$ is the DFT matrix $\rightarrow$ FFT in $\mathcal{O}(N \log N)$
2. **LD Digital Nets + Digitally Shift Invariant (DSI) Kernels**
   - Give Recursive Symmetric Block Toeplitz (RSBT) Gram matrices K
   - $\therefore$ Eigendecomp $\mathsf{K} = \mathsf{V}\Lambda\overline{\mathsf{V}}$ where $\overline{\mathsf{V}}$ is the Hadamard matrix $\rightarrow$ FWHT in $\mathcal{O}(N \log N)$

Fast GPs Pairing LD Points with Special Kernels

1. **LD Lattices + Shift Invariant (SI) Kernels**
   - Give circulant Gram matrices $\mathsf{K} = \{K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})\}_{i,i'=0}^{N-1}$

   $\therefore$ Eigendecomp $\mathsf{K} = \mathsf{V}\Lambda\overline{\mathsf{V}}$ where $\overline{\mathsf{V}}$ is the DFT matrix $\to$ FFT in $\mathcal{O}(N \log N)$
2. **LD Digital Nets + Digitally Shift Invariant (DSI) Kernels**
   - Give Recursive Symmetric Block Toeplitz (RSBT) Gram matrices K

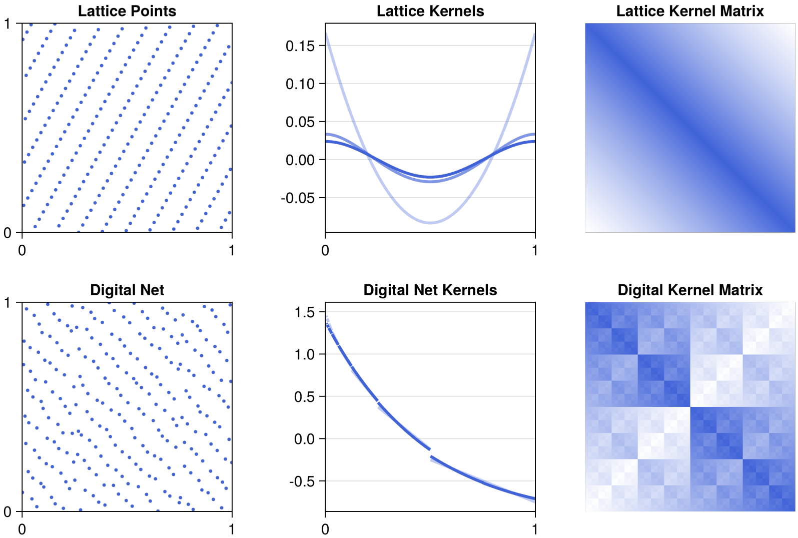
   $\therefore$ Eigendecomp $\mathsf{K} = \mathsf{V}\Lambda\overline{\mathsf{V}}$ where $\overline{\mathsf{V}}$ is the Hadamard matrix $\to$ FWHT in $\mathcal{O}(N \log N)$

Let $\boldsymbol{v}_1 = \mathbf{1}/\sqrt{N}$ and $\boldsymbol{k}_1$ be the first columns of $\overline{\boldsymbol{V}}$ and K respectively:

$$\boldsymbol{\lambda} := \Lambda\mathbf{1} = \sqrt{N}\Lambda\overline{\boldsymbol{v}}_1 = \sqrt{N}\overline{\mathsf{V}}\mathsf{V}\Lambda\overline{\boldsymbol{v}}_1 = \sqrt{N}\overline{\mathsf{V}}\boldsymbol{k}_1$$

- $\mathsf{K}\boldsymbol{a}$, $\mathsf{K}^{-1}\boldsymbol{a}$, and $|\mathsf{K}|$ can all be computed in $\mathcal{O}(N \log N)$ computations
- Only requires evaluating and storing the first column of K

Lattices + SI K = Circulant K and Digital Nets + DSI K = RSBT K



FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.
4. **Batched GPs** for simultaneously modeling vector-output simulations.

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.
4. **Batched GPs** for simultaneously modeling vector-output simulations.
5. **GPU support** enabled by the `PyTorch` stack .

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.
4. **Batched GPs** for simultaneously modeling vector-output simulations.
5. **GPU support** enabled by the `PyTorch` stack .
6. **Flexible LD sequences and SI/DSI kernels** from the Quasi-Monte Carlo Python package `QMCPy` qmcsoftware.github.io/QMCSoftware/.

FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.
4. **Batched GPs** for simultaneously modeling vector-output simulations.
5. **GPU support** enabled by the `PyTorch` stack .
6. **Flexible LD sequences and SI/DSI kernels** from the Quasi-Monte Carlo Python package `QMCPy` qmcsoftware.github.io/QMCSoftware/.
7. **Derivative-informed GPs** for simulations coupled with automatic differentiation.

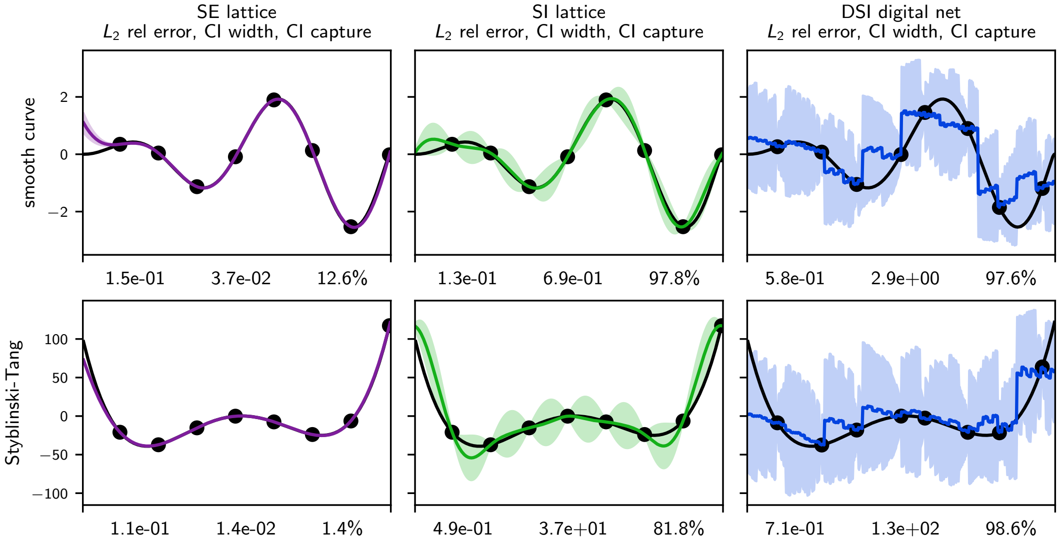
FastGPs Software

`pip install fastgps` or visit alegresor.github.io/fastgps/

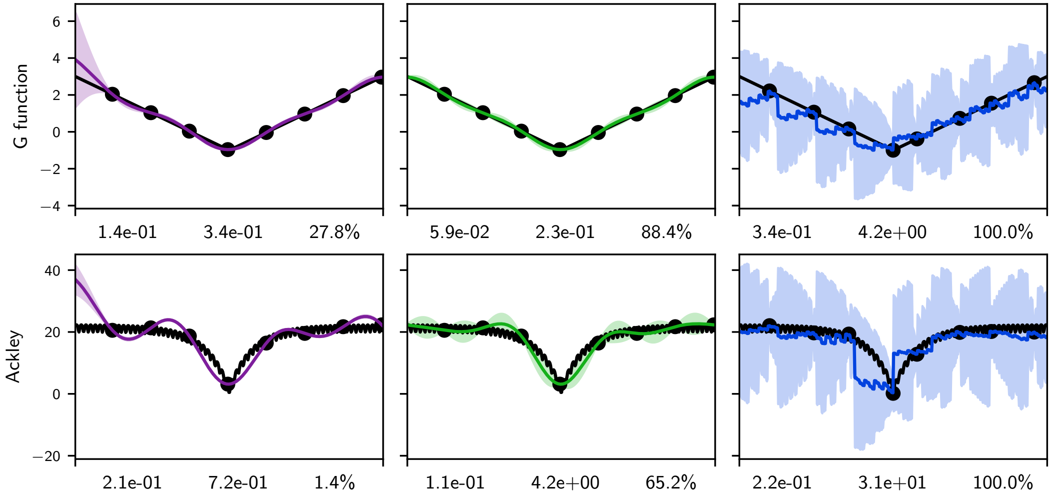
A scalable Python software for fast Gaussian process (GP) regression requiring only

1. **Kernel hyperparameter optimization** of marginal log likelihood (MLL), cross validation (CV), or generalized cross validation (GCV) loss.
2. **Fast Bayesian cubature** for uncertainty quantification in Quasi-Monte Carlo.
3. **Fast multi-task GPs** with support for different sample sizes for each task. Potentially useful for multi-fidelity simulations and Multilevel Monte Carlo.
4. **Batched GPs** for simultaneously modeling vector-output simulations.
5. **GPU support** enabled by the `PyTorch` stack .
6. **Flexible LD sequences and SI/DSI kernels** from the Quasi-Monte Carlo Python package `QMCPy` qmcsoftware.github.io/QMCSoftware/.
7. **Derivative-informed GPs** for simulations coupled with automatic differentiation.
8. **Efficient variance projections** for non-greedy Bayesian optimization in MLMC.

Examples in One Dimension [Surjanovic and Bingham], $N = 8$

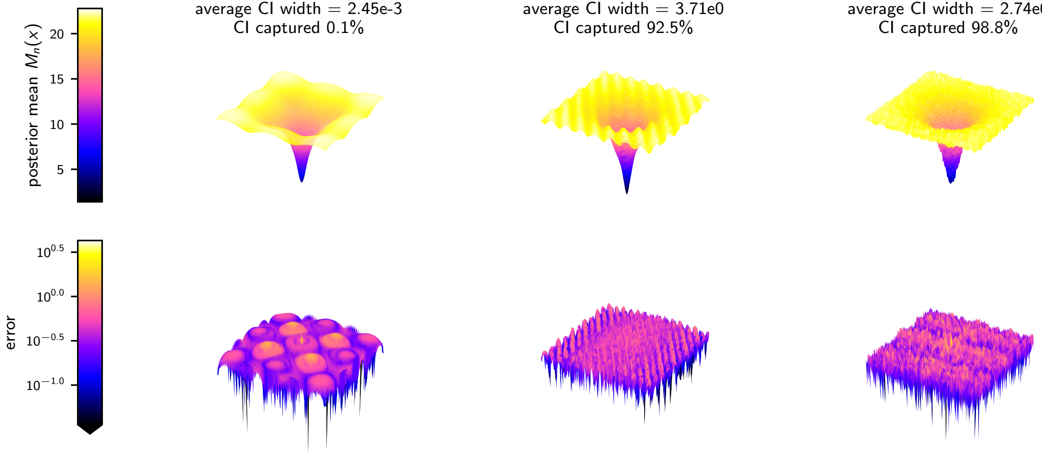


Examples in One Dimension [Surjanovic and Bingham], $N = 8$



Ackley Function in Two Dimensions [Surjanovic and Bingham], $N = 4096$

SE grid	SI lattice	DSI digital net
L_2 relative error = 3.71e-2	L_2 relative error = 2.85e-2	L_2 relative error = 2.62e-2
time = 2.24e0	time = 1.92e-3	time = 4.54e-3
average CI width = 2.45e-3	average CI width = 3.71e0	average CI width = 2.74e0
CI captured 0.1%	CI captured 92.5%	CI captured 98.8%



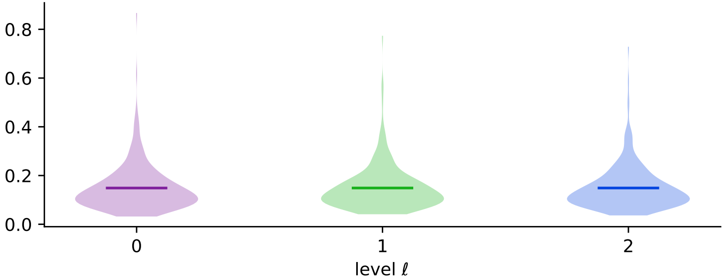
Multilevel (Multi-Task) Modeling

Given a multilevel simulation

$$f:\{1,\dots,L\}\times[0,1]^d\to\mathbb{R},$$

we want to model $f(L,\cdot)$, the true (maximum-fidelity) simulation.

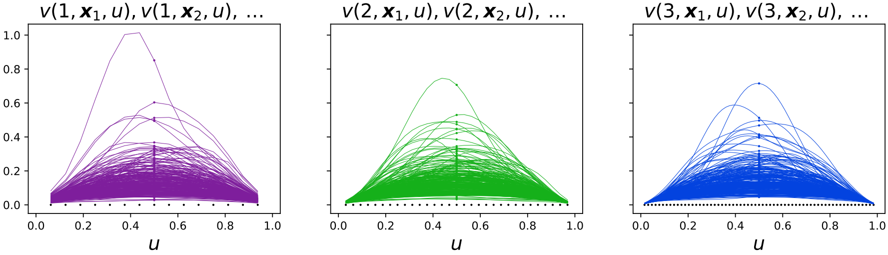
- $f(\ell,\boldsymbol{x})$ simulates at level $\ell\in\{1,\dots,L\}$ and with parameters $\boldsymbol{x}\in[0,1]^d$
- Cost C_ℓ typically greater on higher levels
- $f(1,\cdot),f(2,\cdot),\dots,f(L,\cdot)$ are typically highly correlated



Numerical Solutions of PDEs with Random Coefficients

$$f(\ell,\boldsymbol{x}) = \mathcal{F}(v(\ell,\boldsymbol{x},\cdot))$$

- $v:\{1,\dots,L\}\times[0,1]^d\times\Omega$ is the numerical solution to the PDE
- $\boldsymbol{x}$ represent random coefficients, e.g. coefficients in a Karhunen–Loève expansion
- ℓ controls the fidelity of the numerical solver e.g. the mesh width is $2^{-\ell}$
- $\mathcal{F}$ is a (possibly non-linear) functional of the PDE solution, e.g.,
    - $\mathcal{F}(v(\ell,\boldsymbol{x},\cdot)) = \mathbb{E}[v(\ell,\boldsymbol{x},\boldsymbol{U})]$ where $\boldsymbol{U}\sim\mathcal{U}(\Omega)$, or
    - $\mathcal{F}(v(\ell,\boldsymbol{x},\cdot)) = v(\ell,\boldsymbol{x},u)$ for some $u\in\Omega$, e.g., $u=1/2$ shown below



Multilevel (Quasi-)Monte Carlo without Replications

$$\mathbb{E}[f(L,\boldsymbol{X})] = \sum_{\ell=1}^{L} \mathbb{E}[\underbrace{f(\ell,\boldsymbol{X}) - f(\ell-1,\boldsymbol{X})}_{\Delta_\ell(\boldsymbol{X})}], \quad \boldsymbol{X} \sim \mathcal{U}[0,1]^d, \qquad f(0,\cdot) = 0$$

Multilevel (Quasi-)Monte Carlo without Replications

$$\mathbb{E}[f(L,\boldsymbol{X})] = \sum_{\ell=1}^{L} \mathbb{E}[\underbrace{f(\ell,\boldsymbol{X}) - f(\ell-1,\boldsymbol{X})}_{\Delta_\ell(\boldsymbol{X})}], \quad \boldsymbol{X} \sim \mathcal{U}[0,1]^d, \qquad f(0,\cdot) = 0$$

MLMC $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_i^\ell)$ for IID $\boldsymbol{x}_0^\ell,\dots,\boldsymbol{x}_{N_\ell-1}^\ell \sim \mathcal{U}[0,1]^d$

- $N_\ell \propto \sqrt{\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/C_\ell}$ chosen to optimally minimize error [Giles, 2008]

Multilevel (Quasi-)Monte Carlo without Replications

$$\mathbb{E}[f(L,\boldsymbol{X})] = \sum_{\ell=1}^{L} \mathbb{E}[\underbrace{f(\ell,\boldsymbol{X}) - f(\ell-1,\boldsymbol{X})}_{\Delta_\ell(\boldsymbol{X})}], \quad \boldsymbol{X} \sim \mathcal{U}[0,1]^d, \qquad f(0,\cdot) = 0$$

MLMC $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_i^\ell)$ for IID $\boldsymbol{x}_0^\ell,\dots,\boldsymbol{x}_{N_\ell-1}^\ell \sim \mathcal{U}[0,1]^d$

- $N_\ell \propto \sqrt{\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/C_\ell}$ chosen to optimally minimize error [Giles, 2008]

R-MLQMC $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{R}\sum_{r=1}^{R} \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_{ri}^\ell)$ [Giles and Waterhouse, 2009]

- IID randomizations of low-discrepancy point $\{\boldsymbol{x}_{1i}^\ell\}_{i=0}^{N_\ell-1},\dots,\{\boldsymbol{x}_{Ri}^\ell\}_{i=0}^{N_\ell-1}$.
- N_ℓ greedily doubled on level where $\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/(N_\ell C_\ell)$ the largest
- Variance approximated by sample variance across R sub-means

Multilevel (Quasi-)Monte Carlo without Replications

$$\mathbb{E}[f(L,\boldsymbol{X})] = \sum_{\ell=1}^{L} \mathbb{E}[\underbrace{f(\ell,\boldsymbol{X}) - f(\ell-1,\boldsymbol{X})}_{\Delta_\ell(\boldsymbol{X})}], \quad \boldsymbol{X} \sim \mathcal{U}[0,1]^d, \qquad f(0,\cdot) = 0$$

MLMC $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_i^\ell)$ for IID $\boldsymbol{x}_0^\ell,\dots,\boldsymbol{x}_{N_\ell-1}^\ell \sim \mathcal{U}[0,1]^d$

- $N_\ell \propto \sqrt{\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/C_\ell}$ chosen to optimally minimize error [Giles, 2008]

R-MLQMC $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{R}\sum_{r=1}^{R} \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_{ri}^\ell)$ [Giles and Waterhouse, 2009]

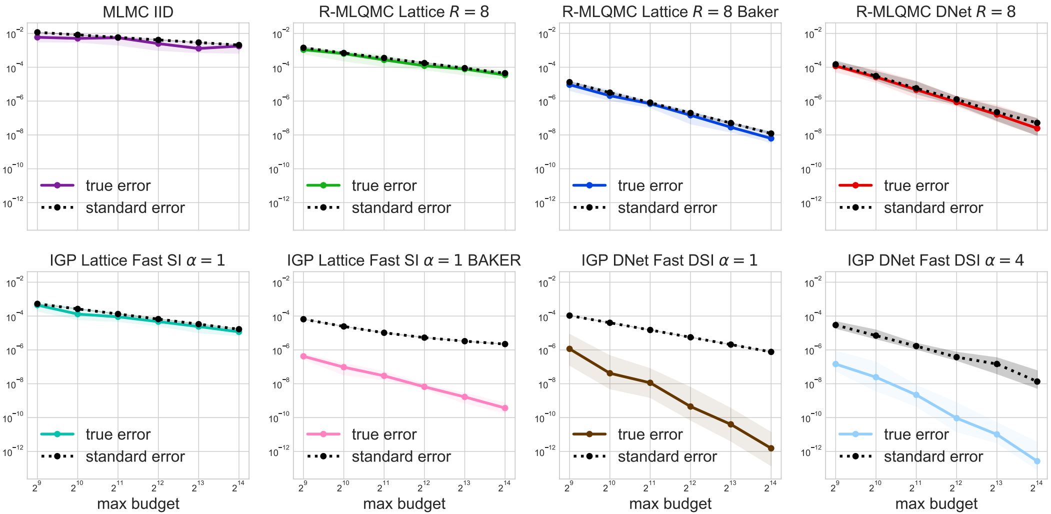
- IID randomizations of low-discrepancy point $\{\boldsymbol{x}_{1i}^\ell\}_{i=0}^{N_\ell-1},\dots,\{\boldsymbol{x}_{Ri}^\ell\}_{i=0}^{N_\ell-1}$.
- N_ℓ greedily doubled on level where $\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/(N_\ell C_\ell)$ the largest
- Variance approximated by sample variance across R sub-means

(IGP) Bayesian MLQMC without replications $\mathbb{E}[\Delta_\ell(\boldsymbol{X})] \approx \frac{1}{N_\ell}\sum_{i=0}^{N_\ell-1} \Delta_\ell(\boldsymbol{x}_i^\ell)$

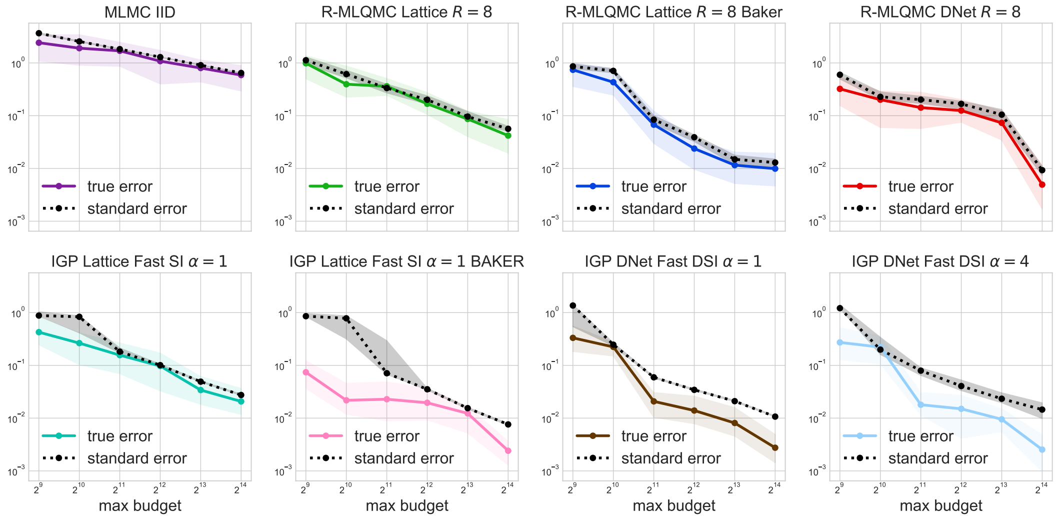
- $\{\boldsymbol{x}_i^\ell\}_{i=0}^{N_\ell-1}$ a single randomized low-discrepancy pointset
- N_ℓ greedily doubled on level where $\mathbb{V}[\Delta_\ell(\boldsymbol{X})]/(N_\ell C_\ell)$ the largest
- Variance taken to be posterior variance with $\Delta_\ell \sim \mathrm{GP}(0,K_\ell)$ (independent GPs)

$$f(\ell,(x_1,x_2)) = \sin(x_1) + 2^{-\ell}\cos(x_2)$$

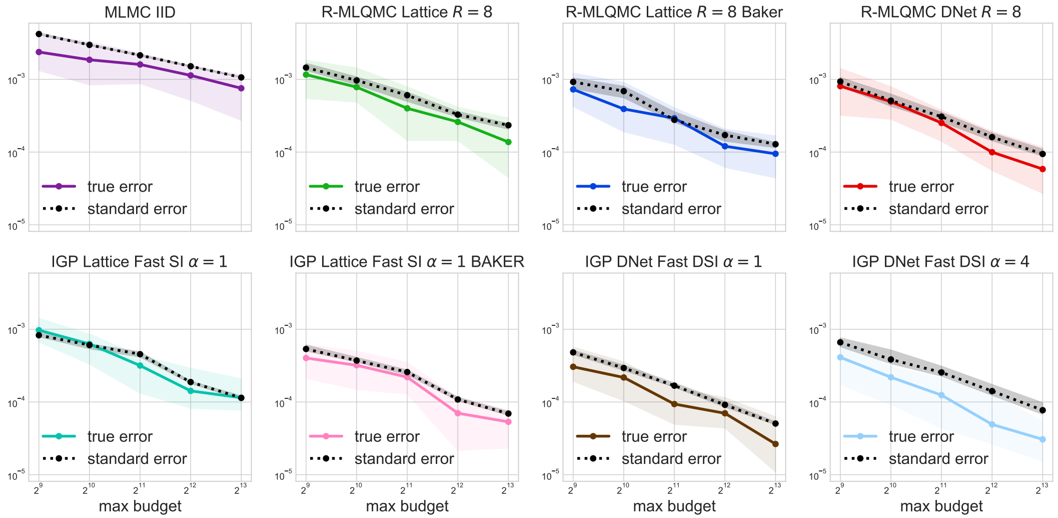
analytic problem with $d = 2$ dimensions and $L = 4$ levels with $T = 50$ trials



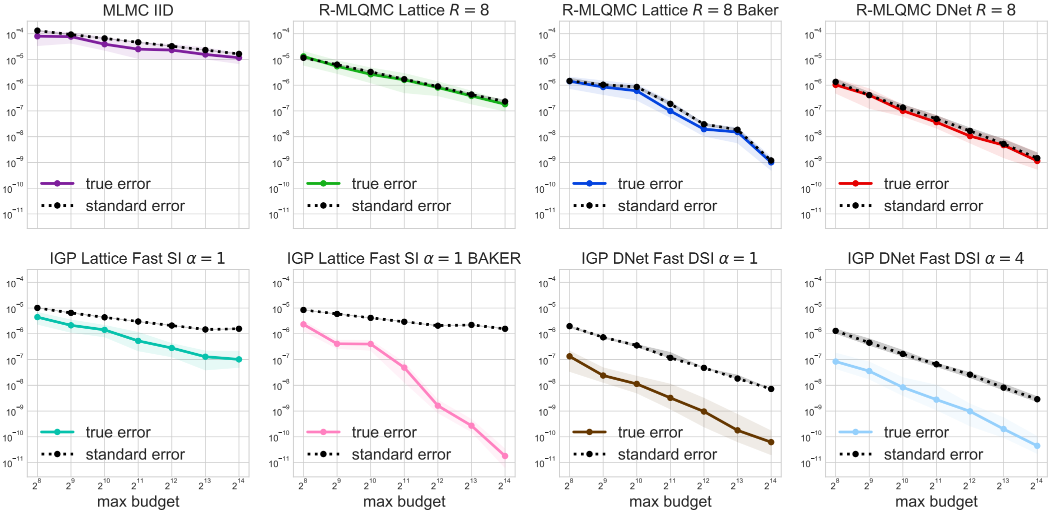
borehole problem with $d = 8$ dimensions and $L = 2$ levels with $T = 50$ trials



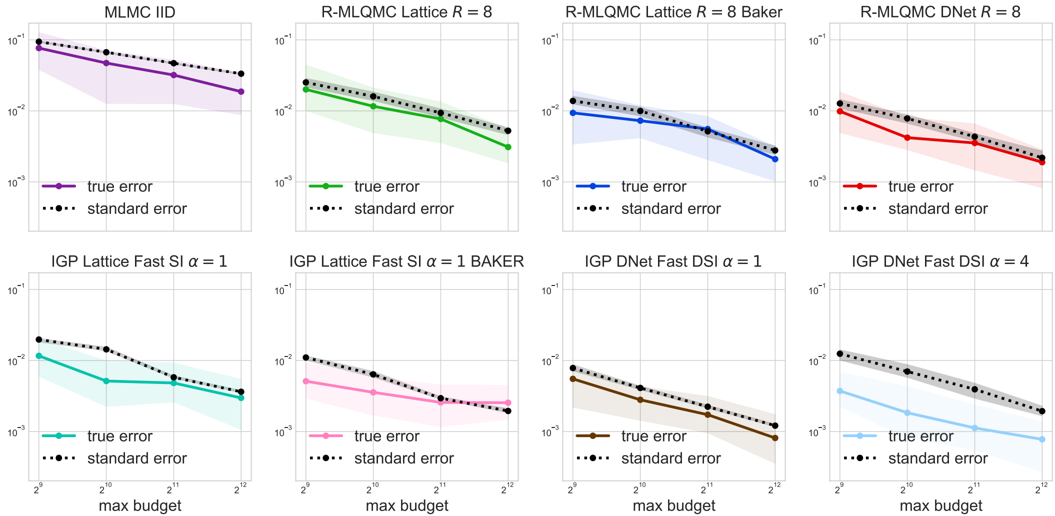
elliptic problem with $d = 8$ dimensions and $L = 4$ levels with $T = 50$ trials



steady state diffusion 1d problem with $d = 9$ dimensions and $L = 5$ levels with $T = 50$ trials



Asian option problem with $d = 16$ dimensions and $L = 8$ levels with $T = 50$ trials



Multi-Task Gaussian Processes

$$f(\ell,\boldsymbol{x}) \sim \mathrm{GP}(0,K)$$

$N = N_1 + \dots + N_L$ sampling locations $\mathcal{D} = \mathcal{D}_1 \cup \dots \cup \mathcal{D}_L$ where $\mathcal{D}_\ell = \left\{\left(\ell, \boldsymbol{x}_i^\ell\right)\right\}_{i=1}^{N_\ell}$.
Posterior mean and covariance

$$\mathbb{E}[f(\ell,\boldsymbol{x})] = \boldsymbol{K}^T(\ell,\boldsymbol{x})\mathsf{K}^{-1}\boldsymbol{f}$$

$$\mathbb{V}[f(\ell,\boldsymbol{x})] = K((\ell,\boldsymbol{x}),(\ell,\boldsymbol{x})) - \boldsymbol{K}^T(\ell',\boldsymbol{x}')\mathsf{K}^{-1}\boldsymbol{K}(\ell,\boldsymbol{x})$$

- $\boldsymbol{K}(\ell,\boldsymbol{x}) = K(\mathcal{D},(\ell,\boldsymbol{x}))$ and $\boldsymbol{f} = f(\mathcal{D})$ are length N vectors
- $\mathsf{K} = K(\mathcal{D},\mathcal{D}^T)$ is the $N \times N$ Gram matrix

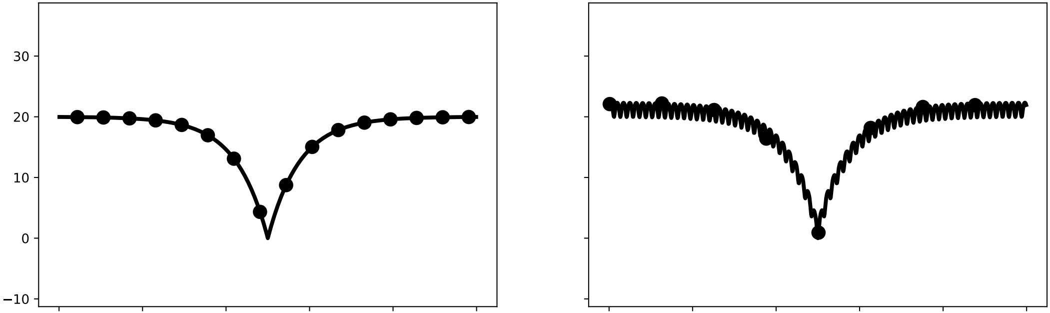
Kernel K depends on hyperparameters $\boldsymbol{\theta}$ e.g. global scale, lengthscale, etc.
Hyperparameters $\boldsymbol{\theta}$ often chosen to minimize negative marginal log likelihood (NMLL)

$$\mathrm{NMLL} \propto \boldsymbol{f}^T\mathsf{K}^{-1}\boldsymbol{f} + \log|\mathsf{K}| + \log(2\pi)N$$

$\therefore$ Multi-Task GPs require computing $\mathsf{K}^{-1}\boldsymbol{f}$ and $\log|\mathsf{K}|$ $\implies$ standard cost $\mathcal{O}(N^3)$

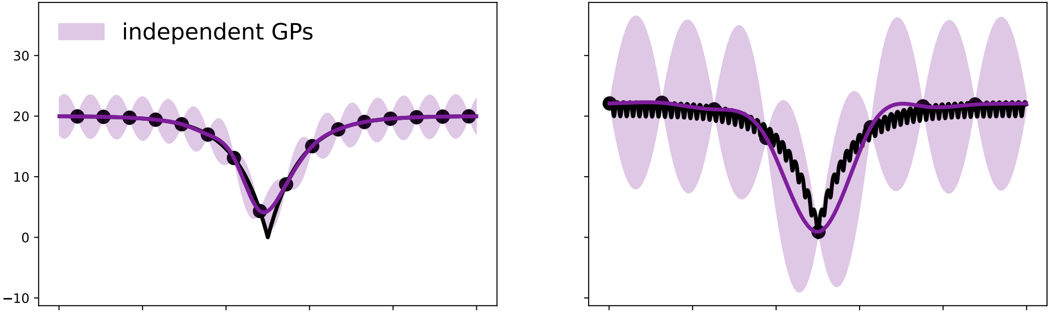
Independent GPs vs a Multi-Task GP Example

Low Fidelity Left, High Fidelity Right



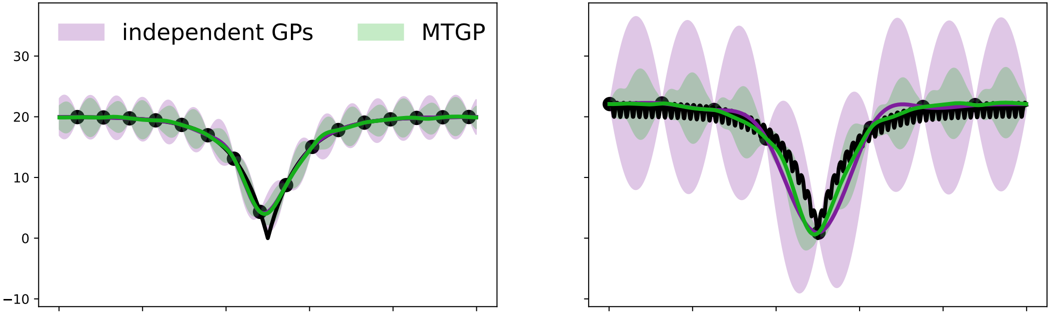
Independent GPs vs a Multi-Task GP Example

Low Fidelity Left, High Fidelity Right



Independent GPs vs a Multi-Task GP Example

Low Fidelity Left, High Fidelity Right



Product Kernels for Multi-Task GPs

Common to assume

$$K((\ell,\boldsymbol{x}),(\ell',\boldsymbol{x}')) = R(\ell,\ell')Q(\boldsymbol{x},\boldsymbol{x}')$$

- $R:\{1,\dots,L\}\times\{1,\dots,L\}\to\mathbb{R}$ an SPD kernel over levels e.g.

$$\mathsf{R} = \{R(\ell,\ell')\}_{\ell,\ell'=1}^{L} = \mathsf{B}\mathsf{B}^T + \mathrm{diag}(\boldsymbol{\nu}), \quad \boldsymbol{\nu}\in\mathbb{R}_+^L, \quad \mathsf{B}\in\mathbb{R}^{L\times r}, \quad \text{rank } r\leqslant L$$

- $Q:[0,1]^d\times[0,1]^d\to\mathbb{R}$ an SPD kernel over parameters

$$\mathsf{K} = \begin{pmatrix} \mathsf{K}_{11} & \cdots & \mathsf{K}_{1L} \\ \vdots & \ddots & \vdots \\ \mathsf{K}_{1L} & \cdots & \mathsf{K}_{LL} \end{pmatrix} = \begin{pmatrix} R_{11}\mathsf{Q}_{11} & \cdots & R_{1L}\mathsf{Q}_{1L} \\ \vdots & \ddots & \vdots \\ R_{1L}\mathsf{Q}_{1L} & \cdots & R_{LL}\mathsf{Q}_{LL} \end{pmatrix}$$

- $R_{\ell\ell'} = R(\ell,\ell')$ is a scalar
- $\mathsf{Q}_{\ell\ell'} = Q(\mathcal{D}_\ell,\mathcal{D}_{\ell'}^T)$ is an $N_\ell\times N_{\ell'}$ Gram matrix

Fast Multi-Task GPs

$$\mathsf{K} = \begin{pmatrix} R_{11}\mathsf{Q}_{11} & \cdots & R_{1L}\mathsf{Q}_{1L} \\ \vdots & \ddots & \vdots \\ \overline{R_{1L}\mathsf{Q}_{1L}} & \cdots & R_{LL}\mathsf{Q}_{LL} \end{pmatrix}$$

Idea: Force "nice" structure in $\mathsf{Q}_{\ell\ell'}$ through special pairings of $\mathsf{X}_\ell = \{\boldsymbol{x}_i^\ell\}_{i=1}^{N_\ell}$ and Q

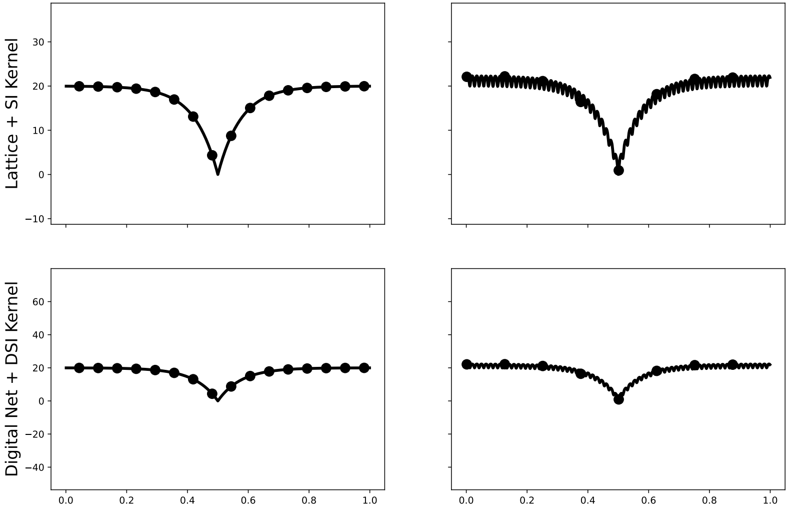
1. X_ℓ a lattice and Q a shift invariant (SI) kernel $\implies$ $\mathsf{Q}_{\ell\ell'}$ block circulant
2. X_ℓ a (base 2) digital net and Q a digitally SI (DSI) kernel $\implies$ $\mathsf{Q}_{\ell\ell}$ block RSBT

Technicalities

- Lattices and digital nets require sample sizes $N_\ell = 2^{m_\ell}$
- Lattices $\mathsf{X}_1,\dots,\mathsf{X}_L$: same generating vector, possibly different random shifts
- Circulant matrices diagonalizable by FFT
- Digital nets $\mathsf{X}_1,\dots,\mathsf{X}_L$: same generating matrices, possibly different digital shifts
- RSBT matrices diagonalizable by FWHT

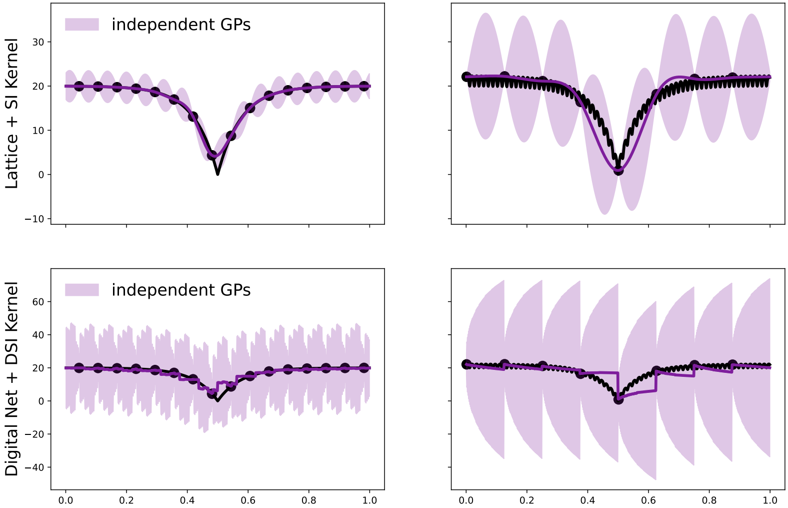
Independent Fast GPs vs Fast Multi-Task GPs Example

Low Fidelity Left, High Fidelity Right



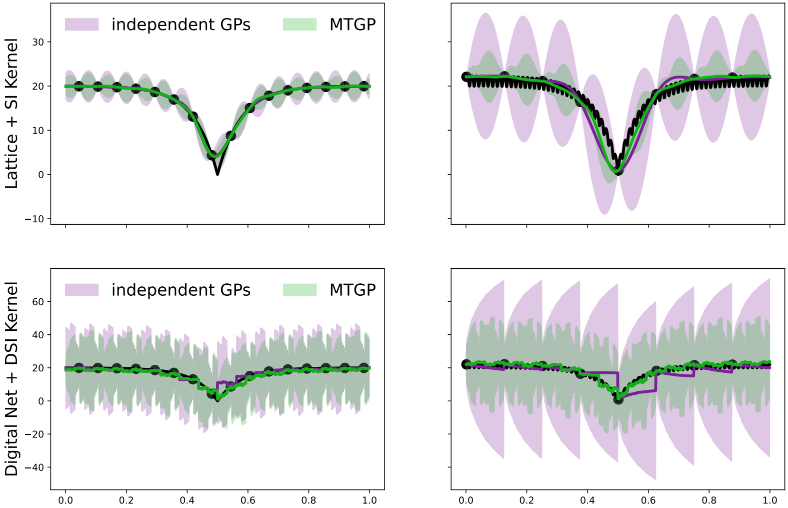
Independent Fast GPs vs Fast Multi-Task GPs Example

Low Fidelity Left, High Fidelity Right



Independent Fast GPs vs Fast Multi-Task GPs Example

Low Fidelity Left, High Fidelity Right



Fast Multi-Task GPs Continued

$$\mathsf{K}_{\ell\ell'} = R_{\ell\ell'}\mathsf{Q}_{\ell\ell'} = \mathsf{V}_{m_\ell}\Sigma_{\ell\ell'}\overline{\mathsf{V}_{m_{\ell'}}}$$

- $\overline{\mathsf{V}_m}$ a $2^m \times 2^m$ *fast transform matrix*
    1. Lattice X_ℓ with SI Q makes $\overline{\mathsf{V}_{m_\ell}}$ the Fast Fourier Transform
    2. Digital Net X_ℓ with DSI Q makes $\overline{\mathsf{V}_{m_\ell}}$ the Fast Walsh Hadamard Transform
- $\mathsf{V}_m\boldsymbol{a}$ and $\overline{\mathsf{V}_m}\boldsymbol{a}$ both cost only $\mathcal{O}(m2^m)$ to compute
- The first column of $\overline{\mathsf{V}_m}$ is $\mathbf{1}_m/\sqrt{2^m}$
- $\Sigma_{\ell\ell'}$ a diagonal block matrix characterized by

$$\boldsymbol{\sigma}_{\ell\ell'} = \Sigma_{\ell\ell'}\mathbf{1}_{m_{\ell'}} = \sqrt{2^{m_{\ell'}}}\overline{\mathsf{V}_{m_\ell}}\boldsymbol{k}_{\ell\ell',1}$$

where $\boldsymbol{k}_{\ell\ell',1}$ is the first column of $\mathsf{K}_{\ell\ell'}$ and we assume $m_\ell \geqslant m_{\ell'}$

Fast Multi-Task GPs NMLL

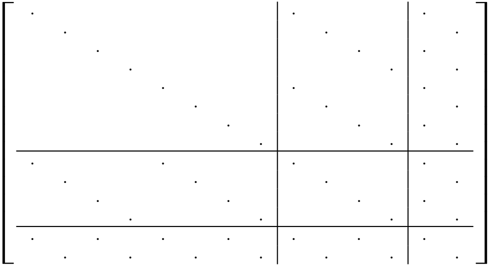
$$\mathsf{K} = \begin{pmatrix} \mathsf{V}_{m_1} & & \\ & \ddots & \\ & & \mathsf{V}_{m_L} \end{pmatrix} \begin{pmatrix} \Sigma_{11} & \cdots & \Sigma_{1L} \\ \vdots & \ddots & \vdots \\ \overline{\Sigma_{1L}} & \cdots & \Sigma_{LL} \end{pmatrix} \begin{pmatrix} \overline{\mathsf{V}_{m_1}} & & \\ & \ddots & \\ & & \overline{\mathsf{V}_{m_L}} \end{pmatrix} =: \mathsf{V}\Sigma\overline{\mathsf{V}}$$

$$\mathrm{NMLL} \propto \widehat{\boldsymbol{f}}^T \Sigma^{-1} \widehat{\boldsymbol{f}} + \log|\Sigma| + \log(2\pi)N, \qquad \widehat{\boldsymbol{f}} = \overline{\mathsf{V}}\boldsymbol{f}$$

$\therefore$ Fast Multi-Task GP fitting requires computing $\Sigma^{-1}\widehat{\boldsymbol{f}}$ and $\log|\Sigma|$

For example, if $N_1 = 8$, $N_2 = 4$, and $N_3 = 2$ then Σ has the following structure

$\Sigma =$



Fast Multi-Task GPs Storage and Costs for Σ^{-1}, $\log\lvert\Sigma\rvert$

- Assume evaluating $f(\ell,\boldsymbol{x})$ costs C_ℓ for any $\boldsymbol{x}\in[0,1]^d$
- Assume evaluating $\mathsf{K}((\ell,\boldsymbol{x}),(\ell',\boldsymbol{x}'))$ costs $\mathcal{O}(d)$
- Assume $m_1 \geqslant \cdots \geqslant m_L$ i.e. less samples on higher levels (or reorder levels)

$$\mathrm{NMLL} \propto \widehat{\boldsymbol{f}}^T \Sigma^{-1}\widehat{\boldsymbol{f}} + \log\lvert\Sigma\rvert + \log(2\pi)N$$

- Evaluate $\boldsymbol{f} = f(\mathcal{D})$ at cost $\mathcal{O}\left(\sum_{\ell=1}^L C_\ell 2^{m_\ell}\right)$
- Evaluate $\widehat{\boldsymbol{f}} = \overline{\mathsf{V}}\boldsymbol{f}$ at cost $\mathcal{O}\left(\sum_{\ell=1}^L m_\ell 2^{m_\ell}\right)$
- Evaluate only first columns of $\mathsf{K}_{\ell\ell'}$ at total cost $\mathcal{O}\left(d\sum_{\ell=1}^L (L-\ell+1)2^{m_\ell}\right)$
- Evaluate Σ at cost $\mathcal{O}\left(\sum_{\ell=1}^L (L-\ell+1)m_\ell 2^{m_\ell}\right)$
- Store Σ in $\mathcal{O}\left(\sum_{\ell=1}^L (L-\ell+1)2^{m_\ell}\right)$ (possibly complex) floats
- Evaluate $\Sigma^{-1}\widehat{\boldsymbol{f}}$ and $\log\lvert\Sigma\rvert$ at cost $\mathcal{O}\left(\sum_{\ell'=1}^L 2^{-m_{\ell'}}\left[\sum_{\ell=1}^{\ell'-1} 2^{m_\ell}\right]^2\right)$ ▸ algorithm

`pip install qmcpy`: qmcsoftware.github.io/QMCSoftware/

- Quasi-Monte Carlo Python software
- LD sequences with randomizations
  - lattices (random shifts)
  - digital nets (digital shifts, LMS, Owen scrambling (NUS), higher-order nets)
  - Halton points (digital shifts, permutation scrambles, NUS)
- Automatic transforms (rewrite problem into $f : [0,1]^d \to \mathbb{R}$)
- Diverse use cases (financial options, parameterized PDEs, sensitivity indices, ...)
- Adaptive stopping criteria (choosing N to meet user-specified error tolerance ε)
- (Digitally)-shift-invariant kernels (of varying smoothness)

`pip install fastgps`: alegresor.github.io/fastgps/

- Single task and multi-task fast GPs (and baseline standard GPs)
- Kernel hyperparameter optimization (MLL, CV, GCV)
- Fast Bayesian cubature (including for multi-task GPs)
- Efficient batched GPs with GPU support

References I

Sangjune Bae, Chanyoung Park, and Nam H Kim. Estimating effect of additional sample on uncertainty reduction in reliability analysis using gaussian process. *Journal of Mechanical Design*, 142(11):111706, 2020.

Pau Batlle, Matthieu Darcy, Bamdad Hosseini, and Houman Owhadi. Kernel methods are competitive for operator learning. *Journal of Computational Physics*, 496: 112549, 2024.

Pau Batlle, Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M Stuart. Error analysis of kernel/gp methods for nonlinear and parametric pdes. *Journal of Computational Physics*, 520:113488, 2025.

François-Xavier Briol, Chris J Oates, Mark Girolami, Michael A Osborne, and Dino Sejdinovic. Probabilistic integration. *Statistical Science*, 34(1):1–22, 2019.

References II

Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M. Stuart. Solving and learning nonlinear pdes with gaussian processes. *Journal of Computational Physics*, 447:110668, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2021.110668. URL https://www.sciencedirect.com/science/article/pii/S0021999121005635.

Yifan Chen, Houman Owhadi, and Florian Schäfer. Sparse cholesky factorization for solving nonlinear pdes via gaussian processes. *Mathematics of Computation*, 2024.

Sou-Cheng T Choi, Yuhan Ding, Fred J Hickernell, Jagadeeswaran Rathinavel, and Aleksei G Sorokin. Challenges in developing great Quasi-Monte Carlo software. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 209–222. Springer, 2022a.

References III

Sou-Cheng T. Choi, Fred J. Hickernell, Rathinavel Jagadeeswaran, Michael J. McCourt, and Aleksei G. Sorokin. Quasi-Monte Carlo software. In Alexander Keller, editor, *Monte Carlo and Quasi-Monte Carlo Methods*, pages 23–47, Cham, 2022b. Springer International Publishing. ISBN 978-3-030-98319-2.

Jon Cockayne, Chris Oates, Tim Sullivan, and Mark Girolami. Probabilistic numerical methods for pde-constrained bayesian inverse problems. In *AIP Conference Proceedings*, volume 1853. AIP Publishing, 2017.

Josef Dick. Higher order scrambled digital nets achieve the optimal rate of the root mean square error for smooth integrands. *The Annals of Statistics*, 39(3):1372 – 1398, 2011. doi: 10.1214/11-AOS880. URL https://doi.org/10.1214/11-AOS880.

Josef Dick and Friedrich Pillichshammer. *Digital nets and sequences: discrepancy theory and quasi–Monte Carlo integration*. Cambridge University Press, 2010.

References IV

Josef Dick, Frances Y Kuo, and Ian H Sloan. High-dimensional integration: the quasi-monte carlo way. *Acta Numerica*, 22:133–288, 2013.

Vincent Dubourg, Bruno Sudret, and Franois Deheeger. Metamodel-based importance sampling for structural reliability analysis. *Probabilistic Engineering Mechanics*, 33: 47–57, 2013.

Peter I. Frazier. A tutorial on bayesian optimization, 2018. URL https://arxiv.org/abs/1807.02811.

Michael Giles. Multilevel monte carlo path simulation. *Operations Research*, 56: 607–617, 06 2008. doi: 10.1287/opre.1070.0496.

Michael B Giles and Benjamin J Waterhouse. Multilevel quasi-monte carlo path simulation. *Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics*, 8:165–181, 2009.

References V

Ivan G Graham, Frances Y Kuo, Dirk Nuyens, Robert Scheichl, and Ian H Sloan. Quasi-monte carlo methods for elliptic pdes with random coefficients and applications. *Journal of Computational Physics*, 230(10):3668–3694, 2011.

Ivan G Graham, Frances Y Kuo, James A Nichols, Robert Scheichl, Ch Schwab, and Ian H Sloan. Quasi-monte carlo finite element methods for elliptic pdes with lognormal random coefficients. *Numerische Mathematik*, 131(2):329–368, 2015.

Fred Hickernell. A generalized discrepancy and quadrature error bound. *Mathematics of computation*, 67(221):299–322, 1998a.

Fred J Hickernell. Lattice rules: how well do they measure up? In *Random and quasi-random point sets*, pages 109–166. Springer, 1998b.

Fred J Hickernell, Lan Jiang, Yuewei Liu, and Art B Owen. Guaranteed conservative fixed width confidence intervals via monte carlo sampling. In *Monte Carlo and Quasi-Monte Carlo Methods 2012*, pages 105–128. Springer, 2013.

References VI

Fred J. Hickernell, Lluís Antoni Jiménez Rugama, and Da Li. Adaptive quasi-monte carlo methods for cubature, 2017.

Fred J Hickernell, Nathan Kirk, and Aleksei G Sorokin. Quasi-monte carlo methods: What, why, and how? *arXiv preprint arXiv:2502.03644*, 2025.

Henrik Wann Jensen, James Arvo, Phil Dutre, Alexander Keller, Art Owen, Matt Pharr, and Peter Shirley. Monte carlo ray tracing. In *ACM SIGGRAPH*, volume 5, page 340769537, 2003.

Vesa Kaarnioja, Yoshihito Kazashi, Frances Y Kuo, Fabio Nobile, and Ian H Sloan. Fast approximation by periodic kernel-based lattice-point interpolation with application in uncertainty quantification. *Numerische Mathematik*, pages 1–45, 2022.

Vesa Kaarnioja, Frances Y Kuo, and Ian H Sloan. Lattice-based kernel approximation and serendipitous weights for parametric pdes in very high dimensions. *arXiv preprint arXiv:2303.17755*, 2023.

References VII

Frances Y Kuo and Dirk Nuyens. Application of quasi-monte carlo methods to elliptic pdes with random diffusion coefficients: a survey of analysis and implementation. *Foundations of Computational Mathematics*, 16:1631–1696, 2016.

Frances Y Kuo, Christoph Schwab, and Ian H Sloan. Quasi-monte carlo finite element methods for a class of elliptic partial differential equations with random coefficients. *SIAM Journal on Numerical Analysis*, 50(6):3351–3374, 2012.

Frances Y Kuo, Christoph Schwab, and Ian H Sloan. Multi-level quasi-monte carlo finite element methods for a class of elliptic pdes with random coefficients. *Foundations of Computational Mathematics*, 15(2):411–449, 2015.

Yongzeng Lai and Jerome Spanier. Applications of monte carlo/quasi-monte carlo methods in finance: option pricing. In *Monte-Carlo and Quasi-Monte Carlo Methods 1998: Proceedings of a Conference held at the Claremont Graduate University, Claremont, California, USA, June 22–26, 1998*, pages 284–295. Springer, 1998.

References VIII

Pierre L'Ecuyer. Quasi-monte carlo methods with applications in finance. *Finance and Stochastics*, 13(3):307–349, 2009.

Pierre L'Ecuyer. Randomized quasi-monte carlo: An introduction for practitioners. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 29–52. Springer, 2016.

Pierre L'Ecuyer, Marvin K Nakayama, Art B Owen, and Bruno Tuffin. Confidence intervals for randomized quasi-Monte Carlo estimators. In *2023 Winter Simulation Conference (WSC)*, pages 445–456. IEEE, 2023.

Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992.

Anthony O'Hagan. Bayes–hermite quadrature. *Journal of statistical planning and inference*, 29(3):245–260, 1991.

References IX

Art B Owen. Randomly permuted (t, m, s)-nets and (t, s)-sequences. In *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing: Proceedings of a conference at the University of Nevada, Las Vegas, Nevada, USA, June 23–25, 1994*, pages 299–317. Springer, 1995.

Art B Owen. Variance and discrepancy with alternative scramblings. *ACM Transactions of Modeling and Computer Simulation*, 13(4), 2003.

Art B. Owen. *Monte Carlo theory, methods and examples*. 2018.

Matthias Raab, Daniel Seibert, and Alexander Keller. Unbiased global illumination with participating media. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 591–605. Springer, 2006.

Rüdiger Rackwitz. Reliability analysis—a review and some perspectives. *Structural safety*, 23(4):365–395, 2001.

References X

Carl Edward Rasmussen and Zoubin Ghahramani. Bayesian Monte Carlo. *Advances in neural information processing systems*, pages 505–512, 2003.

Jagadeeswaran Rathinavel. *Fast automatic Bayesian cubature using matching kernels and designs*. Illinois Institute of Technology, 2019.

Jagadeeswaran Rathinavel and Fred J. Hickernell. Fast automatic bayesian cubature using lattice sampling. *Statistics and Computing*, 29(6):1215–1229, Sep 2019. ISSN 1573-1375. doi: 10.1007/s11222-019-09895-9. URL http://dx.doi.org/10.1007/s11222-019-09895-9.

Jagadeeswaran Rathinavel and Fred J Hickernell. Fast automatic bayesian cubature using sobol' sampling. In *Advances in Modeling and Simulation: Festschrift for Pierre L'Ecuyer*, pages 301–318. Springer, 2022.

S. Ashwin Renganathan, Vishwas Rao, and Ionel M. Navon. Camera: A method for cost-aware, adaptive, multifidelity, efficient reliability analysis, 2022. URL https://arxiv.org/abs/2203.01436.

References XI

Pieterjan Robbe, Dirk Nuyens, and Stefan Vandewalle. A dimension-adaptive multi-index monte carlo method applied to a model of a heat exchanger. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 429–445. Springer, 2016.

Pieterjan Robbe, Dirk Nuyens, and Stefan Vandewalle. A multi-index quasi–monte carlo algorithm for lognormal diffusion problems. *SIAM Journal on Scientific Computing*, 39(5):S851–S872, 2017.

Pieterjan Robbe, Dirk Nuyens, and Stefan Vandewalle. Recycling samples in the multigrid multilevel (quasi-) monte carlo method. *SIAM Journal on Scientific Computing*, 41(5):S37–S60, 2019.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

References XII

Aleksei G. Sorokin and Vishwas Rao. Credible intervals for probability of failure with gaussian processes, 2023.

Aleksei G Sorokin and Jagadeeswaran Rathinavel. On bounding and approximating functions of multiple expectations using quasi-monte carlo. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 583–599. Springer, 2022.

Aleksei G. Sorokin, Aleksandra Pachalieva, Daniel O'Malley, James M. Hyman, Fred J. Hickernell, and Nicolas W. Hengartner. Computationally efficient and error aware surrogate construction for numerical solutions of subsurface flow through porous media. *Advances in Water Resources*, 193:104836, 2024. ISSN 0309-1708. doi: https://doi.org/10.1016/j.advwatres.2024.104836. URL https://www.sciencedirect.com/science/article/pii/S0309170824002239.

References XIII

S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. Retrieved April 9, 2025, from http://www.sfu.ca/~ssurjano.

Carsten Waechter and Alexander Keller. Quasi-monte carlo light transport simulation by efficient ray tracing, May 31 2011. US Patent 7,952,583.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Jian Wu, Saul Toscano-Palmerin, Peter I Frazier, and Andrew Gordon Wilson. Practical multi-fidelity bayesian optimization for hyperparameter tuning. In *Uncertainty in Artificial Intelligence*, pages 788–798. PMLR, 2020.

Andrea Zanette, Junzi Zhang, and Mykel J. Kochenderfer. Robust super-level set estimation using gaussian processes, 2018. URL https://arxiv.org/abs/1811.09977.

References XIV

Xiaoyan Zeng, King-Tai Leung, and Fred J Hickernell. *Error analysis of splines for periodic problems using lattice designs*. Springer, 2006.

Xiaoyan Zeng, Peter Kritzer, and Fred J Hickernell. Spline methods using integration lattices and digital nets. *Constructive Approximation*, 30:529–555, 2009.

Return to slide on **Fast Multi-Task GPs Storage and Costs**

Algorithm 1 Inverse and Determinant of Σ

Require: Σ diagonal block matrix with $m_1 \geqslant \cdots \geqslant m_L$.

$\mathsf{A} \leftarrow \Sigma_{11}^{-1}$ diagonal

$\rho \leftarrow |\Sigma_{11}|$

$\ell \leftarrow 2$

while $\ell \leqslant L$ **do**

$\mathsf{D} \leftarrow \Sigma_{\ell\ell}$ diagonal

$\mathsf{C} \leftarrow \Sigma_{1:\ell-1,\ell}$ diagonal blocks

$\mathsf{S}_\ell \leftarrow \mathsf{D} - \overline{\mathsf{C}}\mathsf{A}^{-1}\mathsf{C}$ diagonal Schur complement

$$\mathsf{A} \leftarrow \begin{pmatrix} \mathsf{A}^{-1} + \mathsf{A}^{-1}\mathsf{C}\mathsf{S}_\ell^{-1}\overline{\mathsf{C}}\mathsf{A}^{-1} & -\mathsf{A}^{-1}\mathsf{C}\mathsf{S}_\ell^{-1} \\ -\mathsf{S}_\ell^{-1}\overline{\mathsf{C}}\mathsf{A}^{-1} & \mathsf{S}_\ell^{-1} \end{pmatrix}$$

$\rho \leftarrow \rho|\mathsf{S}_\ell|$

$\ell \leftarrow \ell + 1$

end while

return $\mathsf{A} = \Sigma^{-1} \quad \text{and} \quad \rho = |\Sigma|$