Monte Carlo methods for approximating failure probability regions

Matteo Croci
(BCAM & Ikerbasque, Bilbao)

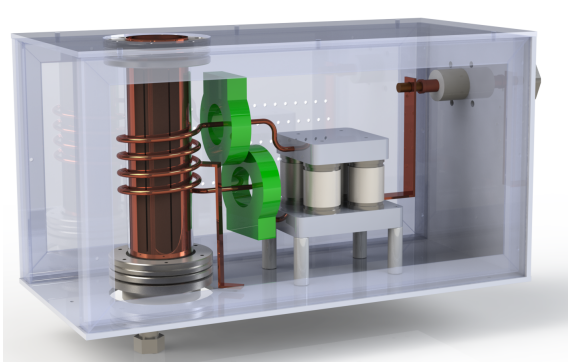
Computational Mathematics and Applications Seminar, University of Oxford, UK, 16 May 2024

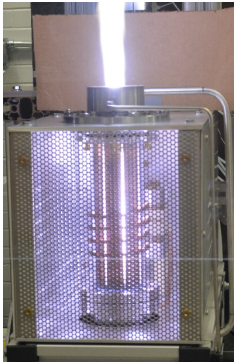
* This material is based upon work supported by the Department of Energy under award number DE-NA0003969.
Visit sponsored by the Marie Skłodowska-Curie Actions programme under grant agreement 101103593.

Overview

1. Introduction

Motivation: Designing inductively coupled plasma torches





Outlook: Designing inductively coupled plasma torches

Challenges:

- Complex multiphysics model. Still under development.
- Prohibitively expensive simulations.
- Large uncertainty in model parameters.

Two algorithms needed:

1. An efficient Monte Carlo method for forward uncertainty quantification.
2. A method for approximating the region of stable torch operating conditions.

Overview

1. Introduction

2. Multilevel best linear unbiased estimators via semidefinite programming

3. Approximating level sets of probability functions

4. Conclusions

2. Multilevel best linear unbiased estimators via semidefinite programming

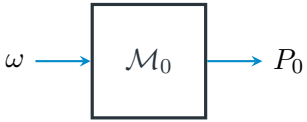
Joint with: K. E. Willcox (UT Austin), S. J. Wright (UW-Madison).

Forward uncertainty propagation and Monte Carlo methods

Forward UQ: given a (computational) *model* that depends on uncertain parameters with known distribution, compute how this uncertainty propagates to the model predictions.

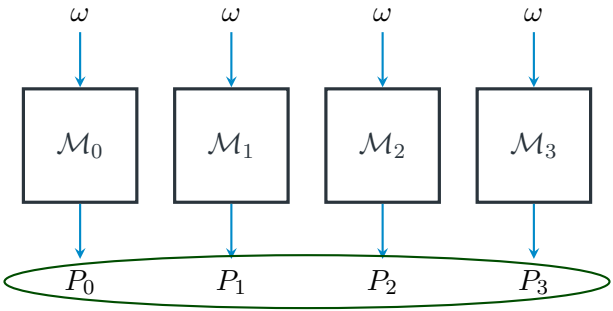
Mathematical formulation: approximate the expectation $\mathbb{E}[P_0]$ of an output QoI $P_0(\omega)$.

Standard approach: Monte Carlo sampling, i.e. $\mathbb{E}[P_0] \approx \frac{1}{n} \sum_{i=1}^{n} P_0(\omega^i)$. Expensive!



Multilevel/multifidelity Monte Carlo methods [Heinrich '01, Giles '08, Ng & Willcox '12, Gorodetsky et al. '20]

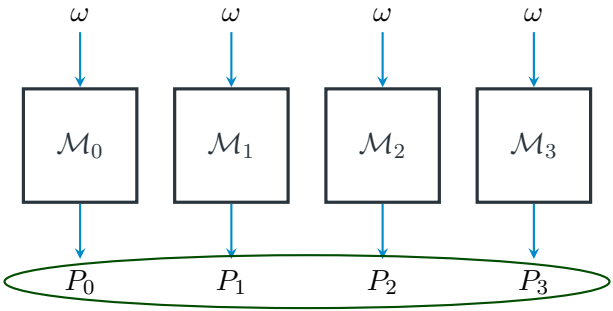
Multilevel and multifidelity Monte Carlo methods strategically combine high- and low-fidelity model samples and exploit their correlations to drastically reduce costs.



Multilevel/multifidelity Monte Carlo methods. $\mathcal{M}_0 =$ high-fidelity model.

Multilevel/multifidelity Monte Carlo methods [Heinrich '01, Giles '08, Ng & Willcox '12, Gorodetsky et al. '20]

Multilevel and multifidelity Monte Carlo methods strategically combine high- and low-fidelity model samples and exploit their correlations to drastically reduce costs.



Multilevel/multifidelity Monte Carlo methods. $\mathcal{M}_0 =$ high-fidelity model.

Note: Different methods combine models differently and in specific groups.

Definition: Model group or combination = set of models sampled with the same input.

Example: Multilevel Monte Carlo [Heinrich '01, Giles '08]

Select L models (model selection) and order them by cost.

$$\mathbb{E}[P_0] = \mathbb{E}[P_{L-1}] + \sum_{\ell=0}^{L-2} \mathbb{E}[P_\ell - P_{\ell+1}].$$

Example: Multilevel Monte Carlo [Heinrich '01, Giles '08]

Select L models (model selection) and order them by cost.

$$\mathbb{E}[P_0] = \mathbb{E}[P_{L-1}] + \sum_{\ell=0}^{L-2} \mathbb{E}[P_\ell - P_{\ell+1}].$$

Apply standard MC to each term on the RHS to obtain the MLMC estimator:

$$\mathbb{E}[P_0] \approx \hat{\mu}_0 = \frac{1}{n_{L-1}} \sum_{i=1}^{n_{L-1}} P_{L-1}(\omega_{L-1}^i) + \sum_{\ell=0}^{L-2} \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} [P_\ell(\omega_\ell^i) - P_{\ell+1}(\omega_\ell^i)].$$

Example: Multilevel Monte Carlo [Heinrich '01, Giles '08]

Select L models (model selection) and order them by cost.

$$\mathbb{E}[P_0] = \mathbb{E}[P_{L-1}] + \sum_{\ell=0}^{L-2} \mathbb{E}[P_\ell - P_{\ell+1}].$$

Apply standard MC to each term on the RHS to obtain the MLMC estimator:

$$\mathbb{E}[P_0] \approx \hat{\mu}_0 = \frac{1}{n_{L-1}} \sum_{i=1}^{n_{L-1}} P_{L-1}(\omega_{L-1}^i) + \sum_{\ell=0}^{L-2} \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} [P_\ell(\omega_\ell^i) - P_{\ell+1}(\omega_\ell^i)].$$

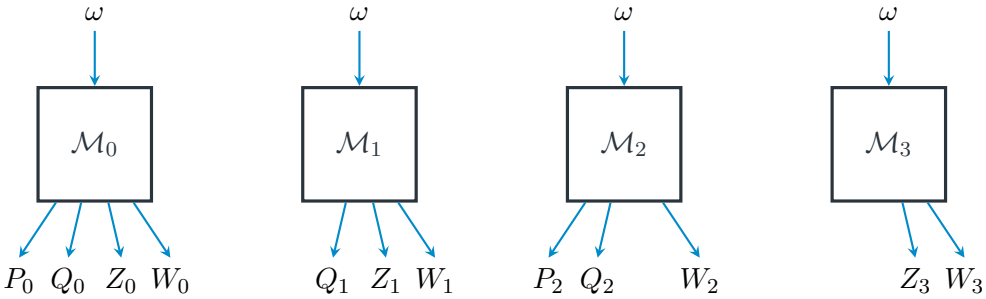
To **find the optimal sample allocation** we must solve the optimization problem

$$\min_{\boldsymbol{n}>0} \mathbb{V}[\hat{\mu}_0] = \sum_{\ell=0}^{L-1} \frac{V_\ell}{n_\ell}, \quad \text{s.t.} \quad \boldsymbol{n}^T \boldsymbol{c} \leq b,$$

For a single-QoI there is a closed-form expression for the optimal $\boldsymbol{n}$.

Model selection and sample allocation problems (MOSAPs)

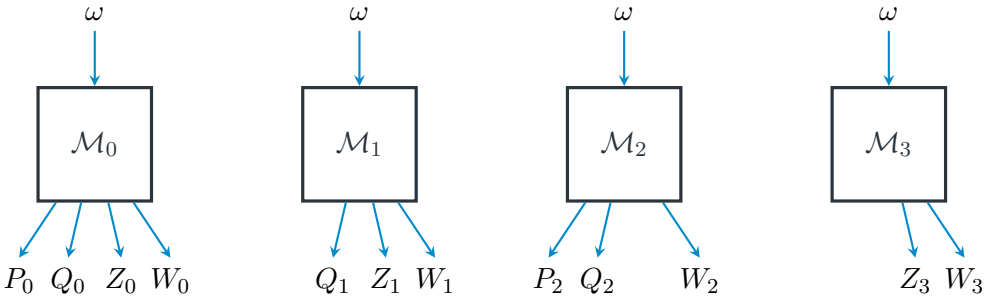
Given a set of available models, how do we find the optimal model combinations?
How many samples should we draw for each model?



All methods require solving a **model selection and sample allocation problem** (**MOSAP**) for their setup. MOSAPs are typically nonlinear, non-convex optimization problems.

Model selection and sample allocation problems (MOSAPs)

Given a set of available models, how do we find the optimal model combinations?
How many samples should we draw for each model?



All methods require solving a **model selection and sample allocation problem** (**MOSAP**) for their setup. MOSAPs are typically nonlinear, non-convex optimization problems.

Objective

Given a set of models and a list of QoIs, automatically construct an optimal estimator.

Multilevel best linear unbiased estimators (MLBLUEs) [Schaden & Ullmann 2020]

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

Multilevel best linear unbiased estimators (MLBLUEs) [Schaden & Ullmann 2020]

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

Consider a single QoI. The MLBLUE algorithm finds an *optimal* estimator $\hat{\boldsymbol{\mu}}$ for $\boldsymbol{\mu} = \mathbb{E}[\boldsymbol{P}]$.

coupled samples of all possible groups $\longrightarrow$ $\boldsymbol{p} = R\boldsymbol{\mu} + \boldsymbol{\varepsilon}$ $\longleftarrow$ error: zero-mean correlated noise

$\uparrow$

mean of $\boldsymbol{p}$

Multilevel best linear unbiased estimators (MLBLUEs) [Schaden & Ullmann 2020]

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

Consider a single QoI. The MLBLUE algorithm finds an *optimal* estimator $\hat{\boldsymbol{\mu}}$ for $\boldsymbol{\mu} = \mathbb{E}[\boldsymbol{P}]$.

coupled samples of all possible groups $\longrightarrow$ $\boldsymbol{p} = R\boldsymbol{\mu} + \boldsymbol{\varepsilon}$ $\longleftarrow$ error: zero-mean correlated noise

$\uparrow$

mean of $\boldsymbol{p}$

The optimal model selection and sample allocation is found by solving the MOSAP

$$\min_{\boldsymbol{n} \geq 0} \mathbb{V}[\boldsymbol{e}^T \hat{\boldsymbol{\mu}}] = \min_{\boldsymbol{n} \geq 0} \boldsymbol{e}^T \Psi^{-1}(\boldsymbol{n}) \boldsymbol{e}, \quad \text{s.t.} \quad \boldsymbol{n}^T \boldsymbol{c} \leq b,$$

$\Psi \succeq 0$ is the matrix of the regression problem for $\boldsymbol{\mu}$, $\boldsymbol{e} = [1, 0, \dots]^T$, $\boldsymbol{n}$ and $\boldsymbol{c}$ are vectors containing the number of samples and costs of each model group respectively, b is the budget.

Multilevel best linear unbiased estimators (MLBLUEs) [Schaden & Ullmann 2020]

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

Consider a single QoI. The MLBLUE algorithm finds an *optimal* estimator $\hat{\boldsymbol{\mu}}$ for $\boldsymbol{\mu} = \mathbb{E}[\boldsymbol{P}]$.

coupled samples of all possible groups $\longrightarrow$ $\boldsymbol{p} = R\boldsymbol{\mu} + \boldsymbol{\varepsilon}$ $\longleftarrow$ error: zero-mean correlated noise

$\uparrow$ mean of $\boldsymbol{p}$

The optimal model selection and sample allocation is found by solving the MOSAP

$$\min_{\boldsymbol{n} \geq 0} \mathbb{V}[\boldsymbol{e}^T \hat{\boldsymbol{\mu}}] = \min_{\boldsymbol{n} \geq 0} \boldsymbol{e}^T \Psi^{-1}(\boldsymbol{n}) \boldsymbol{e}, \quad \text{s.t.} \quad \boldsymbol{n}^T \boldsymbol{c} \leq b,$$

$\Psi \succeq 0$ is the matrix of the regression problem for $\boldsymbol{\mu}$, $\boldsymbol{e} = [1, 0, \dots]^T$, $\boldsymbol{n}$ and $\boldsymbol{c}$ are vectors containing the number of samples and costs of each model group respectively, b is the budget.

Warning: The MLBLUE MOSAP is ill-posed!

MLBLUE extensions required

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

MLBLUE limitations:

1. Requires nonlinear optimization.
2. MOSAP is ill-posed.
3. Standard MLBLUE method is single-output only.

MLBLUE extensions required

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

MLBLUE limitations:

1. Requires nonlinear optimization.
2. MOSAP is ill-posed.
3. Standard MLBLUE method is single-output only.

Our contributions:

1. **Speed and reliability**. Reformulated MOSAP as a semidefinite program for which fast and robust solvers exist. This reformulation also removes ill-conditioning.

MLBLUE extensions required

MLBLUE advantages:

- Optimality.
- In-built automatic model selection strategy.

MLBLUE limitations:

1. Requires nonlinear optimization.
2. MOSAP is ill-posed.
3. Standard MLBLUE method is single-output only.

Our contributions:

1. **Speed and reliability**. Reformulated MOSAP as a semidefinite program for which fast and robust solvers exist. This reformulation also removes ill-conditioning.
2. **Multi-output problems**. Extended to multiple outputs while preserving optimality.

Semidefinite programming reformulation

The MLBLUE standard MOSAP is

$$\min_{\boldsymbol{n}\geq 0} \boldsymbol{e}^T \Psi^{-1}(\boldsymbol{n})\boldsymbol{e}, \quad \text{s.t.} \quad \boldsymbol{n}^T \boldsymbol{c} \leq b.$$

Semidefinite programming reformulation

The MLBLUE standard MOSAP is

$$\min_{\boldsymbol{n}\geq 0} \boldsymbol{e}^T \Psi^{-1}(\boldsymbol{n})\boldsymbol{e}, \quad \text{s.t.} \quad \boldsymbol{n}^T\boldsymbol{c} \leq b.$$

Theorem

The above formulation is equivalent to the following *well-posed* semidefinite program (SDP):

$$\min_{t,\boldsymbol{n}\geq 0} t, \quad \text{s.t.} \quad \Phi(t,\boldsymbol{n}) = \begin{bmatrix} \Psi(\boldsymbol{n}) & \boldsymbol{e} \\ \boldsymbol{e}^T & t \end{bmatrix} \succeq 0, \quad \boldsymbol{n}^T\boldsymbol{c} \leq b, \quad \boldsymbol{n}^T\boldsymbol{h} \geq 1.$$

where $\boldsymbol{h}$ is a known boolean vector. Φ is linear in t and $\Psi(\boldsymbol{n})$, which in turn is linear in $\boldsymbol{n}$.

Multi-output extension: use one semidefinite constraint for each QoI.

Semidefinite programming reformulation

The MLBLUE standard MOSAP is

$$\min_{\boldsymbol{n}\geq 0} \boldsymbol{e}^T \Psi^{-1}(\boldsymbol{n})\boldsymbol{e}, \quad \text{s.t.} \quad \boldsymbol{n}^T\boldsymbol{c} \leq b.$$

Theorem

The above formulation is equivalent to the following *well-posed* semidefinite program (SDP):

$$\min_{t,\boldsymbol{n}\geq 0} \; t, \quad \text{s.t.} \quad \Phi(t,\boldsymbol{n}) = \begin{bmatrix} \Psi(\boldsymbol{n}) & \boldsymbol{e} \\ \boldsymbol{e}^T & t \end{bmatrix} \succeq 0, \quad \boldsymbol{n}^T\boldsymbol{c} \leq b, \quad \boldsymbol{n}^T\boldsymbol{h} \geq 1.$$

where $\boldsymbol{h}$ is a known boolean vector. Φ is linear in t and $\Psi(\boldsymbol{n})$, which in turn is linear in $\boldsymbol{n}$.

Multi-output extension: use one semidefinite constraint for each QoI.

SDPs can be solved almost as reliably and efficiently as a linear program!

Numerical experiments

Hodgkin-Huxley model

The Hodgkin-Huxley model for neuron membrane action potential:

neuron membrane action potential ↓

membrane current: $I(\omega) + \epsilon(\omega)\Delta V$ — potassium current: $g_{\mathsf{K}} n^4 (V_k - V)$ — sodium current: $g_{\mathsf{Na}} m^3 h (V_{\mathsf{Na}} - V)$ — leakage current: $g_l (V_l - V)$

$$
\begin{cases}
C(\omega) V_t &= I(\omega) + \epsilon(\omega)\Delta V + g_{\mathsf{K}} n^4 (V_k - V) + g_{\mathsf{Na}} m^3 h (V_{\mathsf{Na}} - V) + g_l (V_l - V), \\
n_t &= \alpha_n(V)(1-n) - \beta_n(V) n, \quad \longleftarrow \text{potassium gated channel activation} \\
m_t &= \alpha_m(V)(1-m) - \beta_m(V) m, \quad \longleftarrow \text{sodium gated channel activation} \\
h_t &= \alpha_h(V)(1-h) - \beta_h(V) h. \quad \longleftarrow \text{sodium gated channel inactivation}
\end{cases}
$$

Hodgkin-Huxley model

The Hodgkin-Huxley model for neuron membrane action potential:

neuron membrane action potential ↓

$$\begin{cases} C(\omega)V_t &= \overbrace{I(\omega) + \epsilon(\omega)\Delta V}^{\text{membrane current}} + \overbrace{g_{\mathsf{K}} n^4 (V_k - V)}^{\text{potassium current}} + \overbrace{g_{\mathsf{Na}} m^3 h (V_{\mathsf{Na}} - V)}^{\text{sodium current}} + \overbrace{g_l (V_l - V)}^{\text{leakage current}}, \\ n_t &= \alpha_n(V)(1-n) - \beta_n(V) n, & \longleftarrow \text{potassium gated channel activation} \\ m_t &= \alpha_m(V)(1-m) - \beta_m(V) m, & \longleftarrow \text{sodium gated channel activation} \\ h_t &= \alpha_h(V)(1-h) - \beta_h(V) h. & \longleftarrow \text{sodium gated channel inactivation} \end{cases}$$

The corresponding FitzHugh-Nagumo model:

$$\begin{cases} C(\omega)V_t &= I(\omega) + \epsilon(\omega)\Delta V + g_{\mathsf{K}} n^4 (V_k - V) + g_{\mathsf{Na}} m_\infty^3 (\bar{h} - n)(V_{\mathsf{Na}} - V) + g_l (V_l - V), \\ n_t &= \alpha_n(V)(1-n) - \beta_n(V) n, \\ m = m_\infty &= \text{const}, \quad n + h = \bar{h} = \text{const}. & \longleftarrow \text{simplifying assumptions} \end{cases}$$

Hodgkin-Huxley model

The Hodgkin-Huxley model for neuron membrane action potential:

neuron membrane action potential ↓

$$\begin{cases} C(\omega)V_t &= \overbrace{I(\omega)+\epsilon(\omega)\Delta V}^{\text{membrane current}} + \overbrace{g_{\mathsf{K}}n^4(V_k-V)}^{\text{potassium current}} + \overbrace{g_{\mathsf{Na}}m^3h(V_{\mathsf{Na}}-V)}^{\text{sodium current}} + \overbrace{g_l(V_l-V)}^{\text{leakage current}}, \\ n_t &= \alpha_n(V)(1-n)-\beta_n(V)n, & \longleftarrow \text{potassium gated channel activation} \\ m_t &= \alpha_m(V)(1-m)-\beta_m(V)m, & \longleftarrow \text{sodium gated channel activation} \\ h_t &= \alpha_h(V)(1-h)-\beta_h(V)h. & \longleftarrow \text{sodium gated channel inactivation} \end{cases}$$

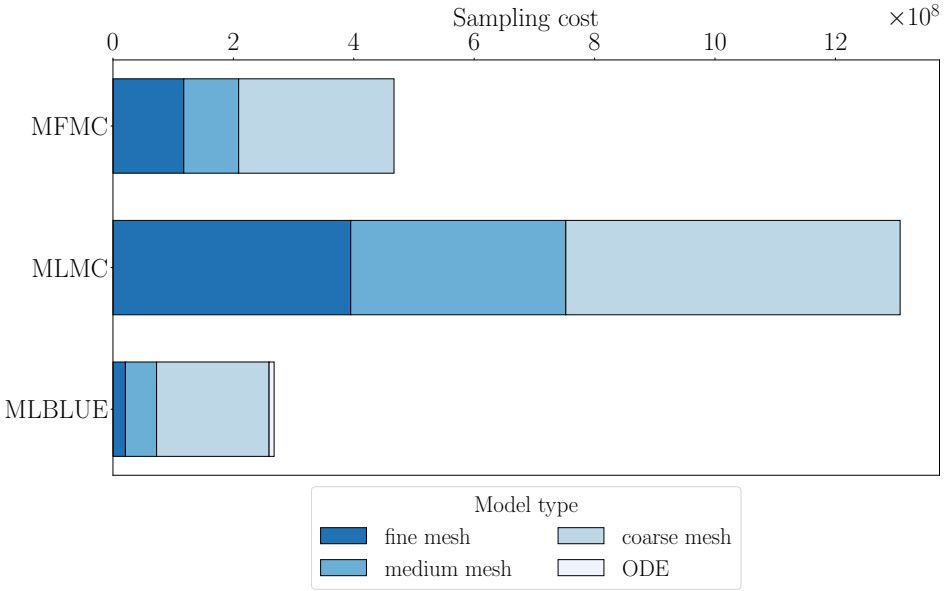
The corresponding FitzHugh-Nagumo model:

$$\begin{cases} C(\omega)V_t &= I(\omega)+\epsilon(\omega)\Delta V + g_{\mathsf{K}}n^4(V_k-V) + g_{\mathsf{Na}}m_\infty^3(\bar{h}-n)(V_{\mathsf{Na}}-V) + g_l(V_l-V), \\ n_t &= \alpha_n(V)(1-n)-\beta_n(V)n, \\ m=m_\infty &= \text{const}, \quad n+h=\bar{h}=\text{const}. & \longleftarrow \text{simplifying assumptions} \end{cases}$$

Domain: unit interval. **BCs:** no-flux on left boundary and zero Dirichlet on the right boundary. **Uncertainty:** diffusivity, capacitance, and current.

QoIs: peak potential, total membrane, ionic, and leakage currents. **Models:** Hodgkin-Huxley and FitzHugh-Nagumo PDEs and ODEs (no diffusion), grid and timestep refinements.

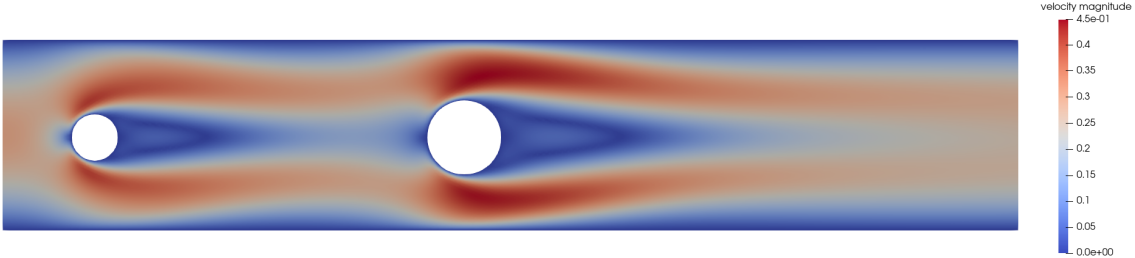
Hodgkin-Huxley model



Note: only MLBLUE uses the FitzHugh-Nagumo models.

Steady Navier-Stokes flow past two cylinders.

$$
\begin{cases}
-\nu(\omega)\Delta \boldsymbol{u} + \boldsymbol{u}\nabla \boldsymbol{u} + \nabla p = 0, \quad \nabla \cdot \boldsymbol{u} = 0, & \boldsymbol{x} \in D_2, & \omega \in \Omega, \\
\boldsymbol{u}|_{\Gamma_t} = \boldsymbol{u}|_{\Gamma_b} = \boldsymbol{u}|_{C_1} = \boldsymbol{u}|_{C_2} = \boldsymbol{0}, & \nu(\omega)\nabla \boldsymbol{u}|_{\Gamma_r} \cdot \boldsymbol{n} - p|_{\Gamma_r}\boldsymbol{n} = 0, & \omega \in \Omega, \\
\boldsymbol{u}|_{\Gamma_l} = \left(\frac{4U(\omega)y(h-y)}{h^2}, 0\right)^T, & \boldsymbol{x} = (x, y)
\end{cases}
$$



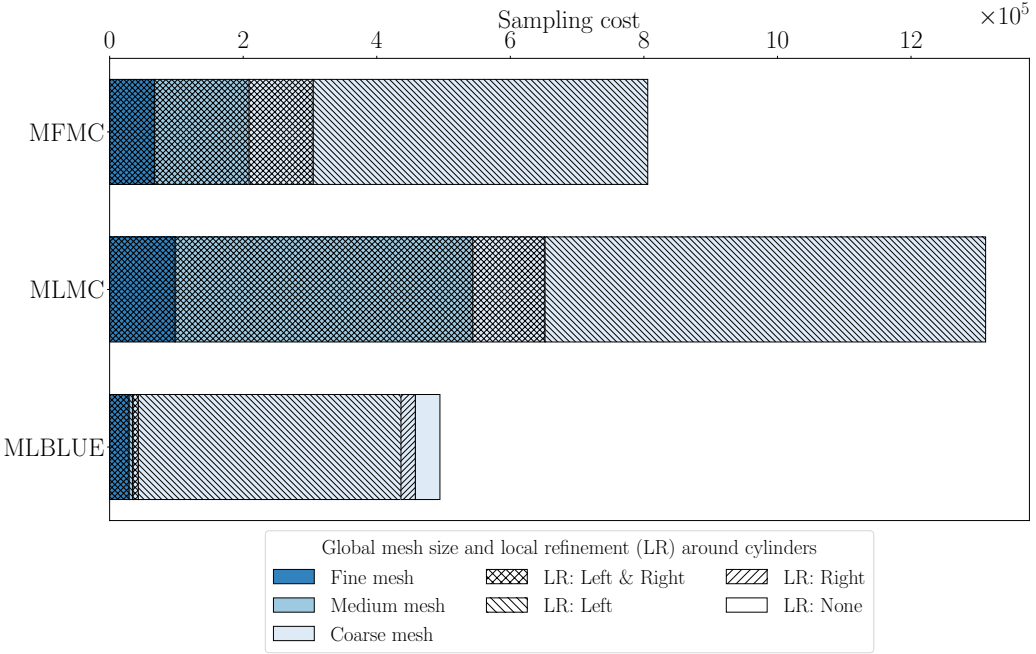
Non-dimensionalized velocity, sample at $\mathrm{Re} = 50$.

Boundaries: Γ_l: left. Γ_r: right. Γ_t: top. Γ_b: bottom. C_1: first cylinder. C_2: second cylinder.

QoIs: lift and drag coefficients and pressure differences at each obstacle.

Models: hierarchy of 3 meshes combined with 4 types of local grid refinements around cylinders (both C_1 and C_2, C_1 only, C_2 only, no local refinement). 12 models in total.

Steady Navier-Stokes flow past two cylinders



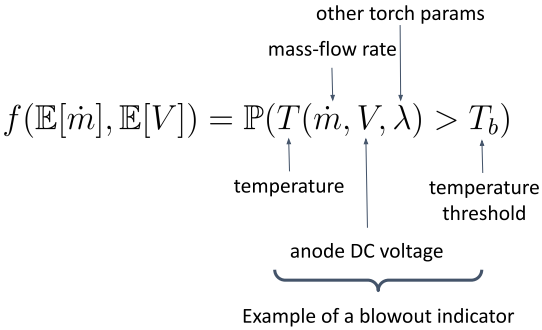
3. Approximating level sets of probability functions

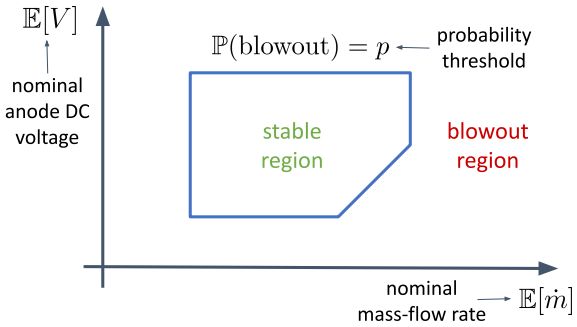
Joint with: A.-L. Haji-Ali (Herriot-Watt University).

Approximating the region of stable torch operating conditions

Background: ICP torches may blow out under some uncertain parameter regimes.

Objective: Find the parameter region in which the torch is stable with high probability.





Approximating level sets of noisy functions - Preliminaries

Mathematical challenge: Approximate the level set of a scalar function $f(\boldsymbol{x})$, $\boldsymbol{x} \in D \subset\subset \mathbb{R}^d$ that is only accessible via expensive and noisy point evaluations (due to Monte Carlo).

Approximating level sets of noisy functions - Preliminaries

Mathematical challenge: Approximate the level set of a scalar function $f(\boldsymbol{x})$, $\boldsymbol{x} \in D \subset\subset \mathbb{R}^d$ that is only accessible via expensive and noisy point evaluations (due to Monte Carlo).

Point evaluations. In general, specialized techniques are required for estimating probabilities (cf. works by Giles, Nobile, Haji-Ali). Here we can use MLBLUE + pre-integration.

Approximating level sets of noisy functions - Preliminaries

Mathematical challenge: Approximate the level set of a scalar function $f(\boldsymbol{x})$, $\boldsymbol{x} \in D \subset\subset \mathbb{R}^d$ that is only accessible via expensive and noisy point evaluations (due to Monte Carlo).

Point evaluations. In general, specialized techniques are required for estimating probabilities (cf. works by Giles, Nobile, Haji-Ali). Here we can use MLBLUE + pre-integration.

Wishlist:

- **Efficiency.** Speedup via spatial and stochastic adaptivity.
- **Robustness and accuracy.** Approximation proven to capture the level set of f with high probability while avoiding spurious level sets.

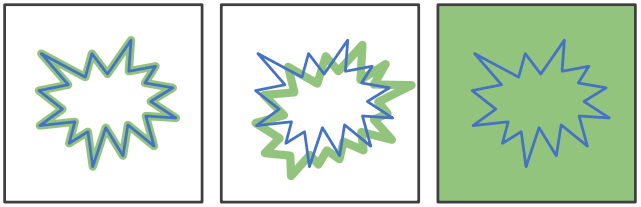
Approximating level sets of noisy functions - Preliminaries

Mathematical challenge: Approximate the level set of a scalar function $f(\boldsymbol{x})$, $\boldsymbol{x} \in D \subset\subset \mathbb{R}^d$ that is only accessible via expensive and noisy point evaluations (due to Monte Carlo).

Point evaluations. In general, specialized techniques are required for estimating probabilities (cf. works by Giles, Nobile, Haji-Ali). Here we can use MLBLUE + pre-integration.

Wishlist:

- **Efficiency.** Speedup via spatial and stochastic adaptivity.
- **Robustness and accuracy.** Approximation proven to capture the level set of f with high probability while avoiding spurious level sets.



Approximating level sets of noisy functions - Preliminaries

Idea: Construct a *goal-oriented adaptive piecewise-polynomial approximation* $\hat{f}(\boldsymbol{x}, \omega)$ of $f(\boldsymbol{x})$ by interpolating Monte Carlo approximations of $f(\boldsymbol{x})$ on an adaptively constructed grid.

Approximating level sets of noisy functions - Preliminaries

Idea: Construct a *goal-oriented adaptive piecewise-polynomial approximation* $\hat{f}(\boldsymbol{x}, \omega)$ of $f(\boldsymbol{x})$ by interpolating Monte Carlo approximations of $f(\boldsymbol{x})$ on an adaptively constructed grid.

Two requirements for adaptivity:

1. ε accuracy on the level set.
2. Just enough accuracy away from the level set so that spurious level sets are avoided.

Approximating level sets of noisy functions - Preliminaries

Idea: Construct a *goal-oriented adaptive piecewise-polynomial approximation* $\hat{f}(\boldsymbol{x}, \omega)$ of $f(\boldsymbol{x})$ by interpolating Monte Carlo approximations of $f(\boldsymbol{x})$ on an adaptively constructed grid.

Two requirements for adaptivity:

1. ε accuracy on the level set.
2. Just enough accuracy away from the level set so that spurious level sets are avoided.

Definitions:

$$L_f(\varepsilon) := \{\boldsymbol{x} \in \bar{D} : \ |f(\boldsymbol{x})| \leq \varepsilon\}, \quad L_{\hat{f}}(\varepsilon, \omega) := \{\boldsymbol{x} \in \bar{D} : \ |\hat{f}(\boldsymbol{x}, \omega)| \leq \varepsilon\}, \quad \text{for } \varepsilon > 0.$$

Approximating level sets of noisy functions - Preliminaries

Idea: Construct a *goal-oriented adaptive piecewise-polynomial approximation* $\hat{f}(\boldsymbol{x}, \omega)$ of $f(\boldsymbol{x})$ by interpolating Monte Carlo approximations of $f(\boldsymbol{x})$ on an adaptively constructed grid.

Two requirements for adaptivity:

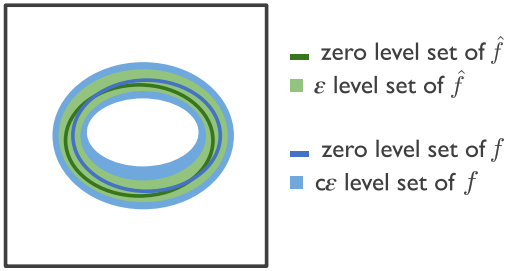
1. ε accuracy on the level set.
2. Just enough accuracy away from the level set so that spurious level sets are avoided.

Definitions:

$$L_f(\varepsilon) := \{\boldsymbol{x} \in \bar{D} : \ |f(\boldsymbol{x})| \leq \varepsilon\}, \quad L_{\hat{f}}(\varepsilon, \omega) := \{\boldsymbol{x} \in \bar{D} : \ |\hat{f}(\boldsymbol{x}, \omega)| \leq \varepsilon\}, \quad \text{for } \varepsilon > 0.$$

Requirement: With high probability, it must hold that $\forall \varepsilon > 0$, $\exists c > 0$,

$$L_f(0) \subseteq L_{\hat{f}}(\varepsilon, \omega) \subseteq L_f(c\varepsilon).$$



Outlook: A work in progress

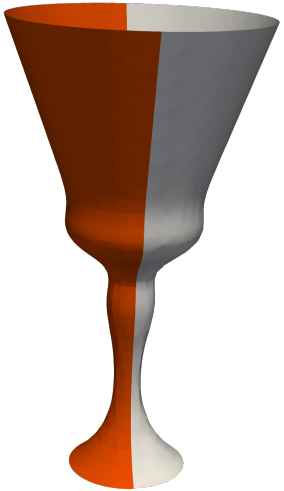
At the moment we have:

- A working convergence and complexity theory in the noise-free case.
- A working algorithm.

What we do not have yet:

- Working theory for noisy evaluations.
- Complete numerical results.

Suggestions are welcome!



approximation
of 3D chalice

Theory for noise-free evaluations

Theorem

$$|f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \leq \max(\varepsilon, |f(\boldsymbol{x})| - \varepsilon) \quad \Longrightarrow \quad L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon).$$

Theory for noise-free evaluations

Theorem

$$|f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \leq \max(\varepsilon, |f(\boldsymbol{x})| - \varepsilon) \quad \Longrightarrow \quad L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon).$$

Main assumption

Let D_h be a mesh of D. For each cell $\square_i \in D_h$ of size h_i, we assume that we can construct an *a posteriori* local error estimator e_i satisfying for $\tilde{c} > 0$, $p_i > 0$,

$$\max_{\boldsymbol{x} \in \square_i} |f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \leq e_i \leq \tilde{c} h_i^{p_i},$$

Theory for noise-free evaluations

Theorem

$$|f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \leq \max(\varepsilon, |f(\boldsymbol{x})| - \varepsilon) \quad \Longrightarrow \quad L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon).$$

Main assumption

Let D_h be a mesh of D. For each cell $\square_i \in D_h$ of size h_i, we assume that we can construct an *a posteriori* local error estimator e_i satisfying for $\tilde{c} > 0$, $p_i > 0$,

$$\max_{\boldsymbol{x} \in \square_i} |f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \leq e_i \leq \tilde{c} h_i^{p_i},$$

Theorem

$$e_i \leq \max\left(\varepsilon, \min_{\boldsymbol{x} \in \square_i} |\hat{f}(\boldsymbol{x})| - \varepsilon\right) \quad \forall i, \quad \Longrightarrow \quad L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon)$$

Algorithm for noise-free problems

$$e_i \leq \max\left(\varepsilon, \min_{\boldsymbol{x}\in\square_i} |\hat{f}(\boldsymbol{x})| - \varepsilon\right) \quad \Longrightarrow \quad L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon) \qquad (\star)$$

Adaptive algorithm

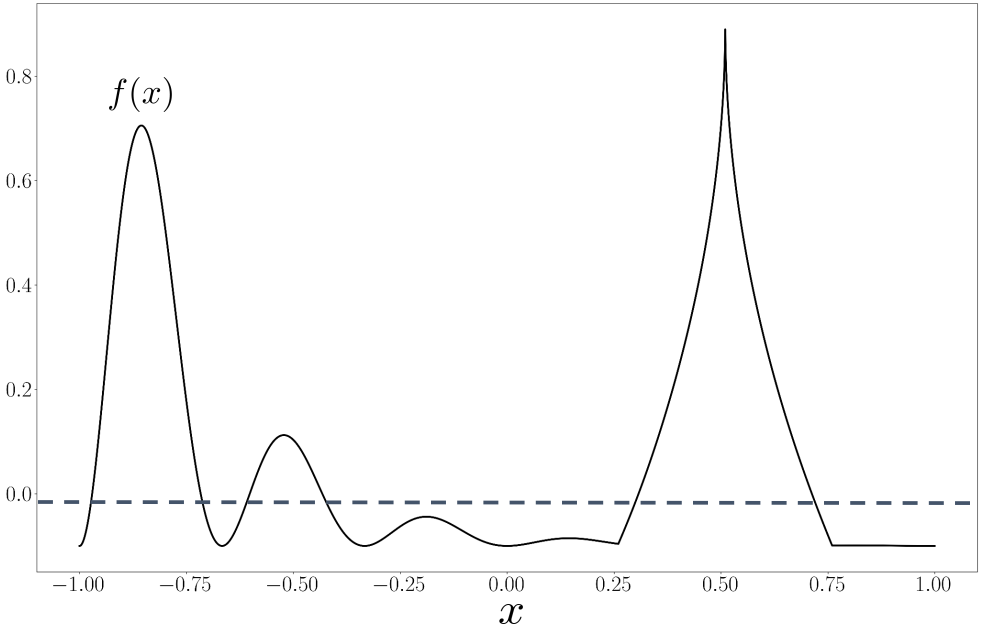
Start from a coarse mesh and construct an initial piecewise polynomial approximation and corresponding local error estimators (we use one round of uniform refinement).

While there exists i violating $(\star)$:

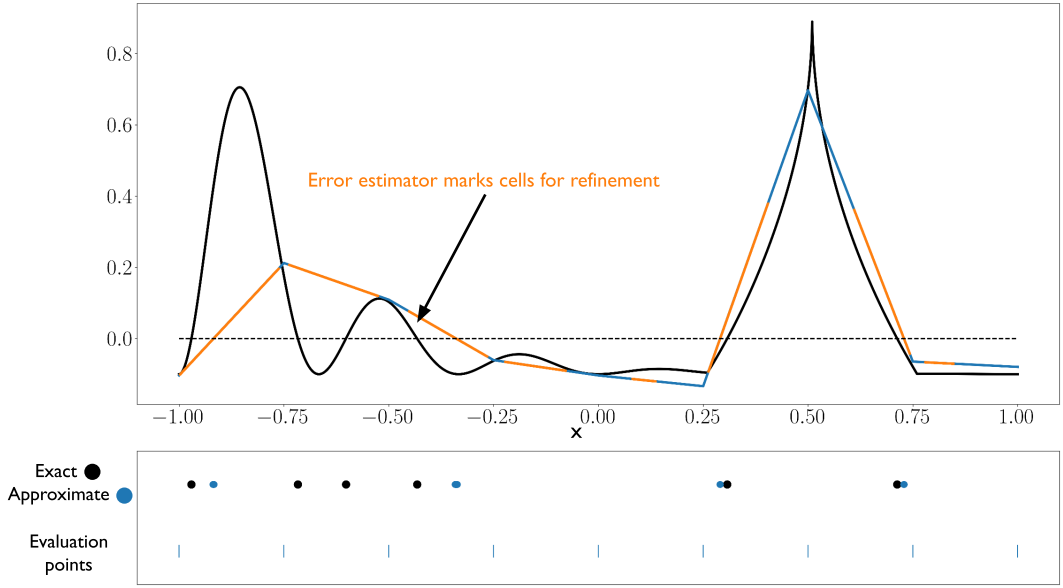
1. Refine all cells violating $(\star)$.
2. Evaluate f at the new mesh nodes.
3. Update the polynomial approximation and the error estimators.

Return the final approximation $\hat{f}$.

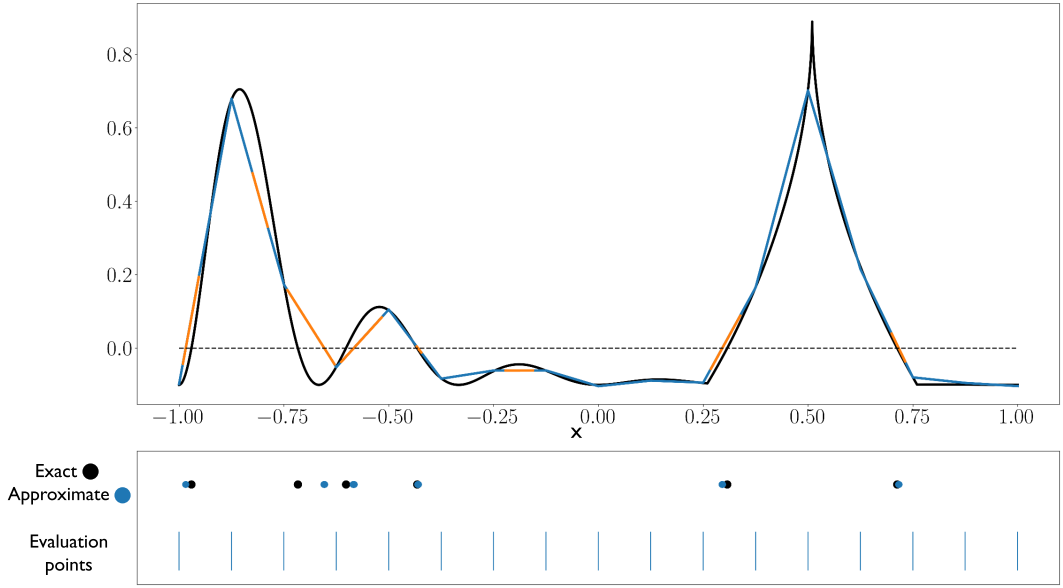
How the algorithm works in practice



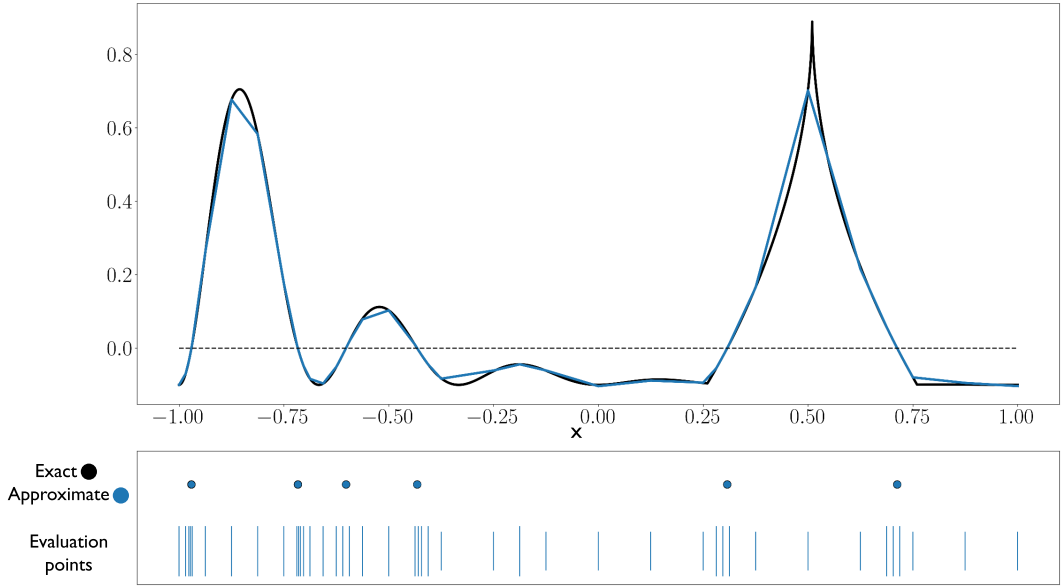
How the algorithm works in practice



How the algorithm works in practice



How the algorithm works in practice



Convergence and complexity theory

Theorem

Let $0 < \varepsilon \ll \delta$ with δ independent from ε and assume that:

1. $f \in C^q(L_f(\delta)) \cup C^s(\bar{D})$ for $q > 1$, $s > 0$.
2. $\nabla f \neq 0$ on $L_f(0)$.

Then the above algorithm finds an approximation $\hat{f}$ satisfying

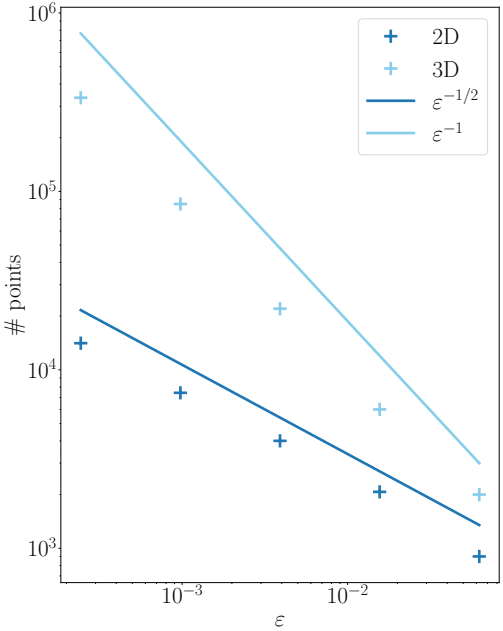
$$L_f(0) \subseteq L_{\hat{f}}(\varepsilon) \subseteq L_f(2\varepsilon)$$

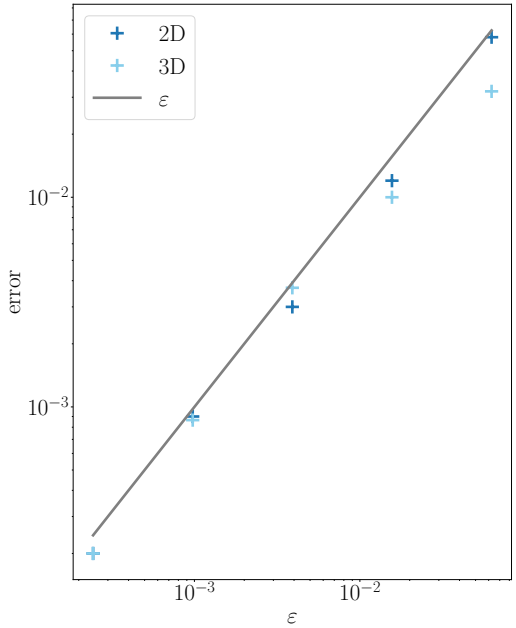
with an asymptotic cost complexity bounded by

$$\mathcal{C}_{\text{tot}} \leq \begin{cases} c_p c_{\text{eval}} \max\left(1,\ \log\left|\log(\varepsilon)\right|\right), & d = 1, \\ c_p c_{\text{eval}} \varepsilon^{\frac{1-d}{p}}, & d > 1, \end{cases}$$

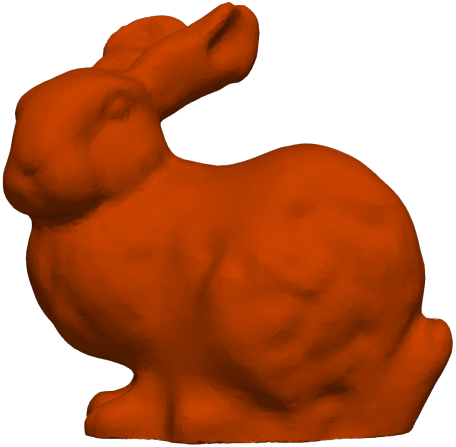
where $c_p > 0$, and $c_{\text{eval}} > 0$ is the maximum cost of a function evaluation.

Complexity and Accuracy - inverse quartic (2D) + chalice (3D)

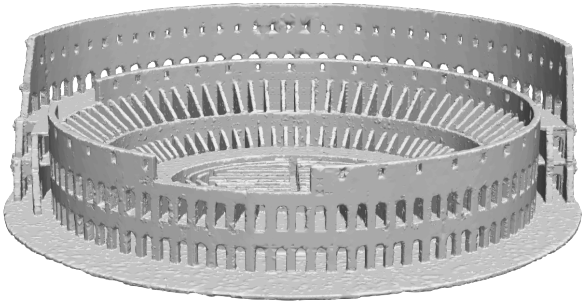




Numerical results - 3D Level-set estimation with noise



1 million evaluations



10 million evaluations

Numerical results - Hyperelastic beam with uncertain Lamé parameters

$$\min_u \int_D \psi(u,\omega)\,\mathrm{d}x - \int_D B\cdot u\,\mathrm{d}x - \int_{\partial D} T\cdot u\,\mathrm{d}x$$

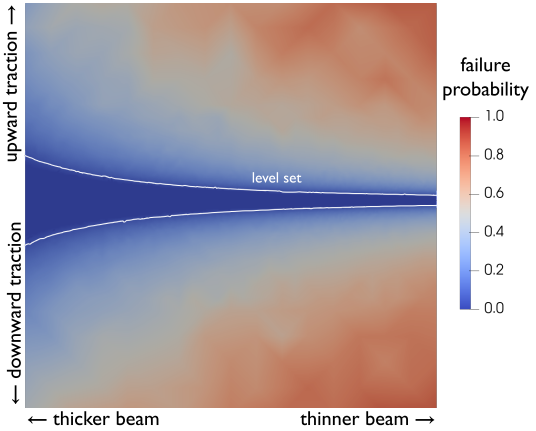
Uncertain elastic stored energy density model

Body force (gravity)

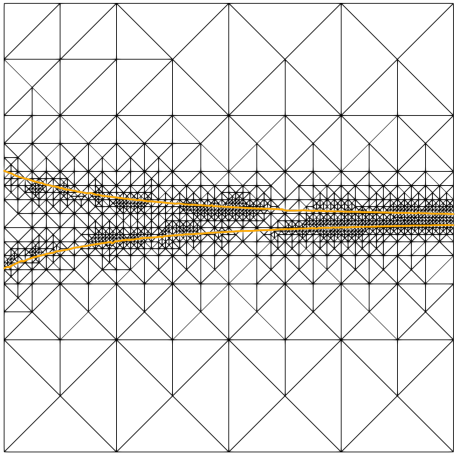
Traction

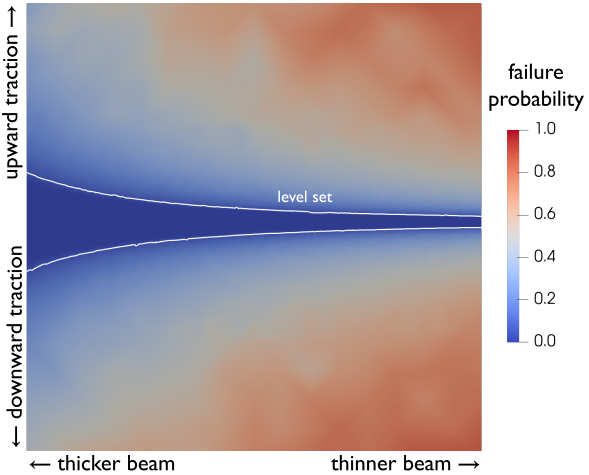
$$f(\theta) = \mathbb{P}(\text{failure}) = \mathbb{P}(\|u\|_{L^\infty(D)} > u_\text{f})$$

θ = { beam thickness, traction }



Numerical results - Hyperelastic beam with uncertain Lamé parameters





4. Conclusions

Conclusions

To sum up

- MLBLUEs are powerful and their automatism is appealing to computational engineers, but their efficiency is tied to how accurately their MOSAPs can be solved.
- The new SDP MOSAPs can be solved reliably and efficiently and extend to the multi-output case. We obtained 100x MOSAP speedup.
- Approximating level sets of probability functions is a challenging problem. Our adaptive strategy is efficient and proven to converge.
- For noisy evaluations, the theory extends as-is by simply replacing $|\cdot|$ with $\mathbb{E}[|\cdot|]$. The challenge is in the construction of a robust cell-wise estimator.

MLBLUE open-source software
`github.com/croci/bluest`

Thank you for listening!

More info about me and my work at: `croci.github.io`
MLBLUE open-source software: `github.com/croci/bluest`

[1] M. Croci and A.-L. Haji-Ali. Level-set approximation of noisy functions and probability density functions. *In preparation*, 2024.

[2] M. Croci, K. E. Willcox, and S. J. Wright. Multi-output multilevel best linear unbiased estimators via semidefinite programming. *Computer Methods in Applied Mechanics and Engineering*, 413:116130, 2023.

[3] M. B. Giles. MLMC techniques for discontinuous functions. *arXiv preprint arXiv:2301.02882*, 2023.

[4] A.-L. Haji-Ali, J. Spence, and A. L. Teckentrup. Adaptive multilevel Monte Carlo for probabilities. *SIAM Journal on Numerical Analysis*, 60(4):2125–2149, 2022.

[5] M. B. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56(3):607–617, 2008. doi: 10.1287/opre.1070.0496.

[6] M. B. Giles. Multilevel Monte Carlo methods. *Acta Numerica*, 24:259–328, 2015. doi: 10.1017/S09624929.

[7] L. W. Ng and K. E. Willcox. Multifidelity approaches for optimization under uncertainty. *International Journal for numerical methods in Engineering*, 100(10):746–772, 2014.

[8] D. Schaden and E. Ullmann. On multilevel best linear unbiased estimators. *SIAM/ASA Journal on Uncertainty Quantification*, 8(2):601–635, 2020.

[9] S. Krumscheid and F. Nobile. Multilevel Monte Carlo approximation of functions. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):1256–1293, 2018.