mimclib

Abdul-Lateef Haji-Ali

Mathematical Institute, University of Oxford.

August 1, 2017

Outline

The central question

Given an $\mathbb{R}^n$-valued random variable, $\boldsymbol{X}$ with PDF $f(\boldsymbol{x}) : \mathbb{R}^n \to [0, \infty)$ and a function, $g : \mathbb{R}^n \to \mathbb{R}$, assume that we are interested in computing the quantity

$$\mathrm{E}[g(\boldsymbol{X})] = \int_{\mathbb{R}^n} g(\boldsymbol{x}) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}.$$

The central question

Given an $\mathbb{R}^n$-valued random variable, $\boldsymbol{X}$ with PDF $f(\boldsymbol{x}) : \mathbb{R}^n \to [0, \infty)$ and a function, $g : \mathbb{R}^n \to \mathbb{R}$, assume that we are interested in computing the quantity

$$\mathrm{E}[g(\boldsymbol{X})] = \int_{\mathbb{R}^n} g(\boldsymbol{x}) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}.$$

Possible difficulties:

- PDF, f, is inaccessible, only (approximate) samples of $\boldsymbol{X}$ can be generated.
  - E.g., $\boldsymbol{X}$ is the solution of an SDE.
- Dimension, n, is large or even infinite, hence the integral is expensive to approximate.
  - E.g., expansion of a random field.

Setup

Our objective is to build an estimator $\mathcal{A} \approx \mathrm{E}[g(\boldsymbol{X})]$ with minimal work where

$$P(|\mathcal{A} - \mathrm{E}[g(\boldsymbol{X})]| \leq \mathrm{TOL}) \geq \epsilon$$

for a given accuracy TOL and a given confidence level determined by $0 < \epsilon < 1$.

Setup

Our objective is to build an estimator $\mathcal{A} \approx \mathrm{E}[g(\boldsymbol{X})]$ with minimal work where

$$P(|\mathcal{A} - \mathrm{E}[g(\boldsymbol{X})]| \leq \mathrm{TOL}) \geq \epsilon$$

for a given accuracy TOL and a given confidence level determined by $0 < \epsilon < 1$.
Instead, we impose the following, more restrictive, two constraints:

Bias constraint: $$|\mathrm{E}[\mathcal{A} - g(\boldsymbol{X})]| \leq (1-\theta)\mathrm{TOL},$$

Statistical constraint: $$P\left(|\mathcal{A} - \mathrm{E}[\mathcal{A}]| \leq \theta\mathrm{TOL}\right) \geq \epsilon.$$

For some tolerance splitting, $\theta \in (0, 1)$.

Setup

Our objective is to build an estimator $\mathcal{A} \approx \mathrm{E}[g(\boldsymbol{X})]$ with minimal work where

$$P(|\mathcal{A} - \mathrm{E}[g(\boldsymbol{X})]| \leq \mathrm{TOL}) \ \geq \epsilon$$

for a given accuracy TOL and a given confidence level determined by $0 < \epsilon < 1$.
Instead, we impose the following, more restrictive, two constraints:

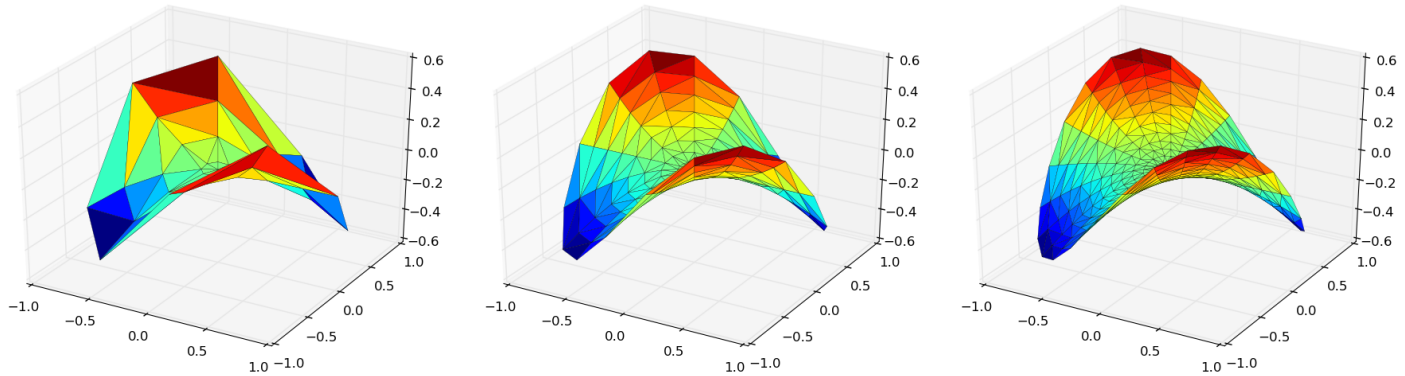
$$\textbf{Bias constraint:} \qquad |\mathrm{E}[\mathcal{A} - g(\boldsymbol{X})]| \leq (1-\theta)\mathrm{TOL},$$

$$\textbf{Statistical constraint:} \qquad \mathrm{Var}[\mathcal{A}] \leq \left(\frac{\theta \mathrm{TOL}}{\Phi^{-1}\left(\frac{\epsilon+1}{2}\right)} \right)^2 .$$

For some tolerance splitting, $\theta \in (0,1)$. Assuming (at least asymptotic) normality of the estimator, $\mathcal{A}$. Here, Φ^{-1} is the inverse of the standard normal CDF.

Numerical approximation

Notation: g_ℓ for $\ell \in \mathbb{N}$ is the approximation of g calculated based on discretization parameters $\boldsymbol{h}_\ell = (h_{\ell,i})_{i=1}^d$.



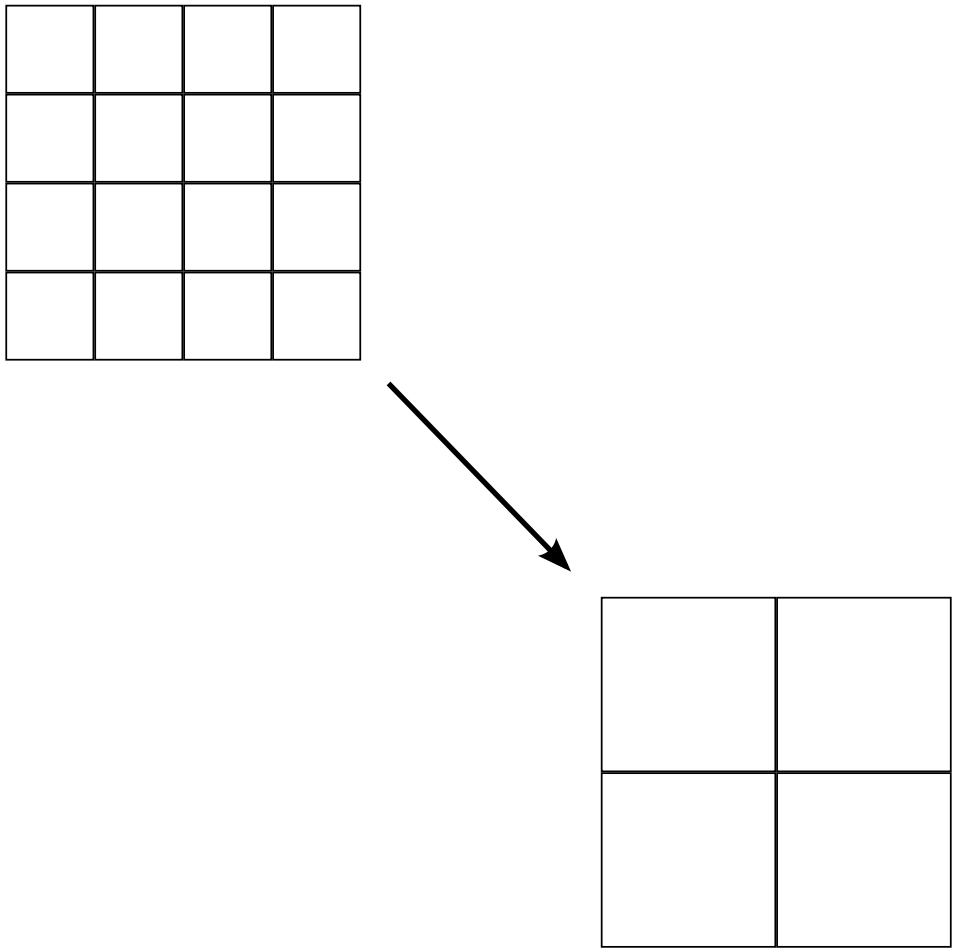
Monte Carlo

The simplest (and most popular) estimator is the Monte Carlo estimator

$$\mathcal{A}_{\mathrm{MC}}[g_L; M] = \frac{1}{M} \sum_{m=1}^{M} g_L(\boldsymbol{X}^{(m)}),$$

for a given level, L, and number of samples, M, that we can choose to satisfy the error constraints and minimize the work.

MLMC main idea: Variance reduction



Multilevel Monte Carlo (MLMC) (Heinrich, 1998) and (Giles, 2008)

Observe the telescopic identity

$$\mathrm{E}[g] \approx \mathrm{E}[g_L] = \mathrm{E}[g_0] + \sum_{\ell=1}^{L} \mathrm{E}[g_\ell - g_{\ell-1}]$$

Multilevel Monte Carlo (MLMC) (Heinrich, 1998) and (Giles, 2008)

Observe the telescopic identity

$$\mathrm{E}[g] \approx \mathrm{E}[g_L] = \mathrm{E}[g_0] + \sum_{\ell=1}^{L} \mathrm{E}[g_\ell - g_{\ell-1}] = \sum_{\ell=0}^{L} \mathrm{E}[\Delta_\ell g],$$

where

$$\Delta_\ell g = \begin{cases} g_0 & \text{if } \ell = 0, \\ g_\ell - g_{\ell-1} & \text{if } \ell > 0. \end{cases}$$

Multilevel Monte Carlo (MLMC) (Heinrich, 1998) and (Giles, 2008)

Observe the telescopic identity

$$\mathrm{E}[g] \approx \mathrm{E}[g_L] = \mathrm{E}[g_0] + \sum_{\ell=1}^{L} \mathrm{E}[g_\ell - g_{\ell-1}] = \sum_{\ell=0}^{L} \mathrm{E}[\Delta_\ell g],$$

where

$$\Delta_\ell g = \begin{cases} g_0 & \text{if } \ell = 0, \\ g_\ell - g_{\ell-1} & \text{if } \ell > 0. \end{cases}$$

Then, using Monte Carlo to approximate each difference independently, the MLMC estimator can be written as

$$\mathcal{A}_{\mathrm{MLMC}}[g; L] = \sum_{\ell=0}^{L} \mathcal{A}_{\mathrm{MC}}[\Delta g_\ell; M_\ell].$$

Main idea: variance reduction using cheaper approximations.

Assumptions

Assumption MC1 (Bias): $|\mathrm{E}[g - g_\ell]| \lesssim \exp(-w\ell)$,
Assumption MC2 (Work): $\mathrm{Work}[g_\ell] \lesssim \exp(d\gamma\ell)$,
Assumption MLMC3 (Variance): $\mathrm{Var}[g_\ell - g_{\ell-1}] \lesssim \exp(-s\ell)$

for all $\ell \in \mathbb{N}$ and positive constants γ, w, s.

Assumptions

Assumption MC1 (Bias): $|\mathrm{E}[g - g_\ell]| \lesssim \exp(-w\ell)$,
Assumption MC2 (Work): $\mathrm{Work}[g_\ell] \lesssim \exp(d\gamma\ell)$,
Assumption MLMC3 (Variance): $\mathrm{Var}[g_\ell - g_{\ell-1}] \lesssim \exp(-s\ell)$

for all $\ell \in \mathbb{N}$ and positive constants γ, w, s.

The optimal work of MLMC is

$$\begin{cases} \mathcal{O}\left(\mathrm{TOL}^{-2}\right) & s > d\gamma \\ \mathcal{O}\left(\mathrm{TOL}^{-2}\right)\left(\log\left(\mathrm{TOL}^{-1}\right)\right)^2 & s = d\gamma \\ \mathcal{O}\left(\mathrm{TOL}^{-2-\frac{d\gamma-s}{w}}\right) & s < d\gamma \end{cases}$$

Assumptions

Assumption MC1 (Bias): $|\mathrm{E}[g - g_\ell]| \lesssim \exp(-w\ell)$,
Assumption MC2 (Work): $\mathrm{Work}[g_\ell] \lesssim \exp(d\gamma\ell)$,
Assumption MLMC3 (Variance): $\mathrm{Var}[g_\ell - g_{\ell-1}] \lesssim \exp(-s\ell)$

for all $\ell \in \mathbb{N}$ and positive constants γ, w, s.

Recall: Optimal cost of Monte Carlo is $\mathcal{O}\left(\mathrm{TOL}^{-2-\frac{d\gamma}{w}}\right)$

The optimal work of MLMC is

$$\begin{cases} \mathcal{O}\left(\mathrm{TOL}^{-2}\right) & s > d\gamma \\ \mathcal{O}\left(\mathrm{TOL}^{-2}\right)\left(\log\left(\mathrm{TOL}^{-1}\right)\right)^2 & s = d\gamma \\ \mathcal{O}\left(\mathrm{TOL}^{-2-\frac{d\gamma-s}{w}}\right) & s < d\gamma \end{cases}$$

Notice: the cost exponentially increases with increasing d.

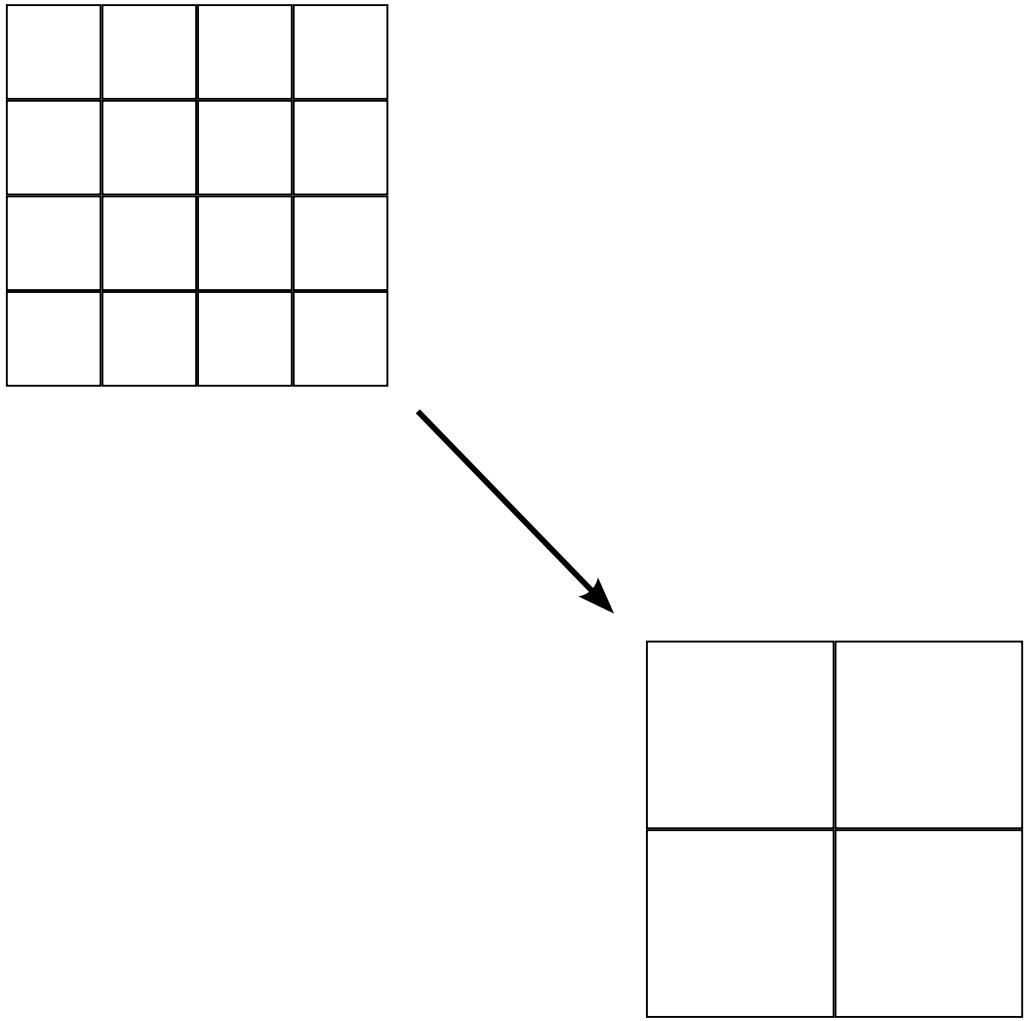
Continuation MLMC

- To compute M_ℓ we need to find L and find estimates of V_ℓ.

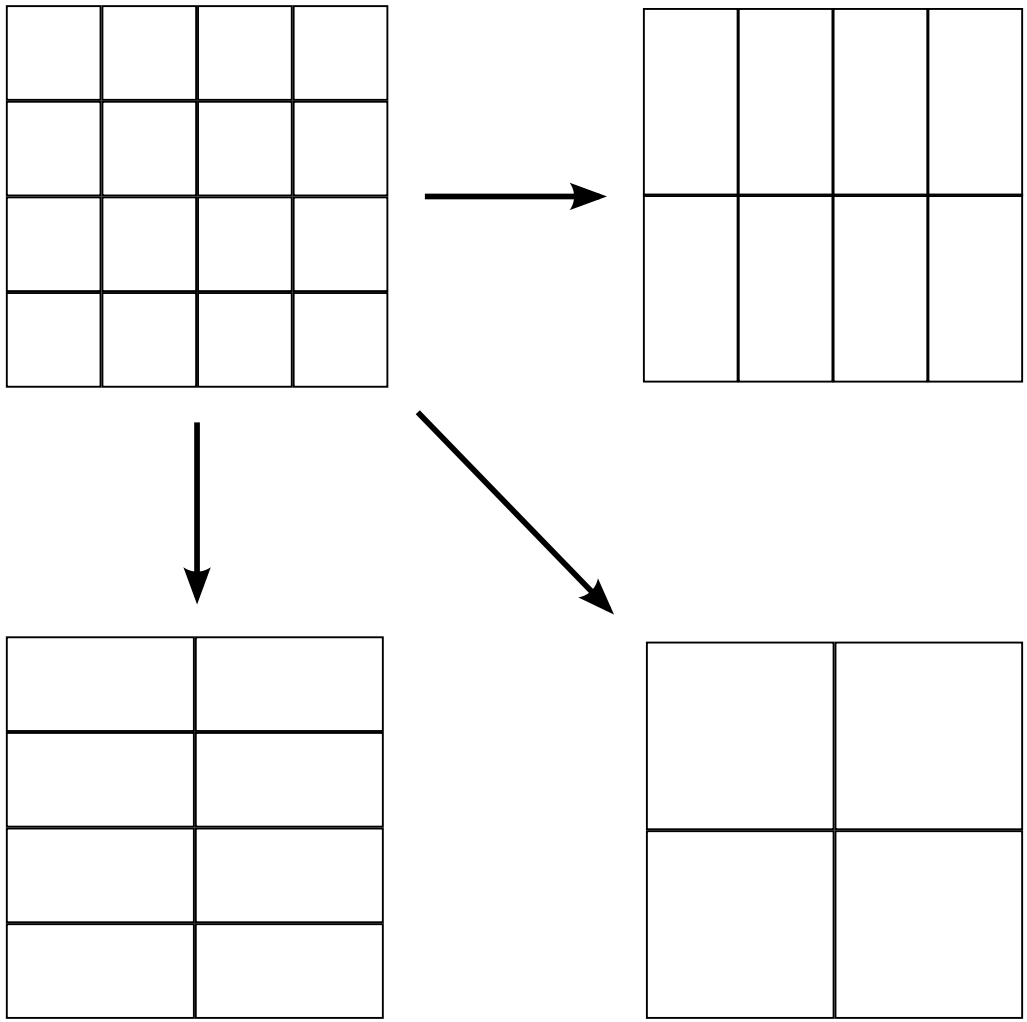
Continuation MLMC

- To compute M_ℓ we need to find L and find estimates of V_ℓ.
- Instead of running for the small required TOL, CMLMC runs a sequence of MLMC realisations, for decreasing tolerances, ending with the required TOL.
- In each step, estimates of V_ℓ are generated using a Bayesian setting which uses **Assumption MLMC3** coupled with the generated samples to produce good estimates even with a small number of samples.
- The value of L is also chosen in each step to minimize the work. This allows choosing a better splitting parameter, θ.
- CMLMC does not have to reuse samples between iterations, ensuring an unbiased estimator for level L approximation.

Variance reduction: MLMC



Variance reduction: Further potential



MIMC Estimator

Consider discretization parameters possibly different in each direction

$$h_{i,\alpha_i} = h_{i,0}\,\beta_i^{-\alpha_i}$$

with $\beta_i > 1$. For a multi-index $\boldsymbol{\alpha} = (\alpha_i)_{i=1}^d \in \mathbb{N}^d$, we denote by $g_{\boldsymbol{\alpha}}$ the approximation of g calculated using a discretization defined by $\boldsymbol{\alpha}$.

MIMC Estimator

Consider discretization parameters possibly different in each direction

$$h_{i,\alpha_i} = h_{i,0}\,\beta_i^{-\alpha_i}$$

with $\beta_i > 1$. For a multi-index $\boldsymbol{\alpha} = (\alpha_i)_{i=1}^d \in \mathbb{N}^d$, we denote by $g_{\boldsymbol{\alpha}}$ the approximation of g calculated using a discretization defined by $\boldsymbol{\alpha}$.

For $i = 1, \ldots, d$, define the first order difference operators

$$\Delta_i g_{\boldsymbol{\alpha}} = \begin{cases} g_{\boldsymbol{\alpha}} & \text{if } \alpha_i = 0, \\ g_{\boldsymbol{\alpha}} - g_{\boldsymbol{\alpha}-\boldsymbol{e}_i} & \text{if } \alpha_i > 0, \end{cases}$$

MIMC Estimator

Construct the first order mixed difference

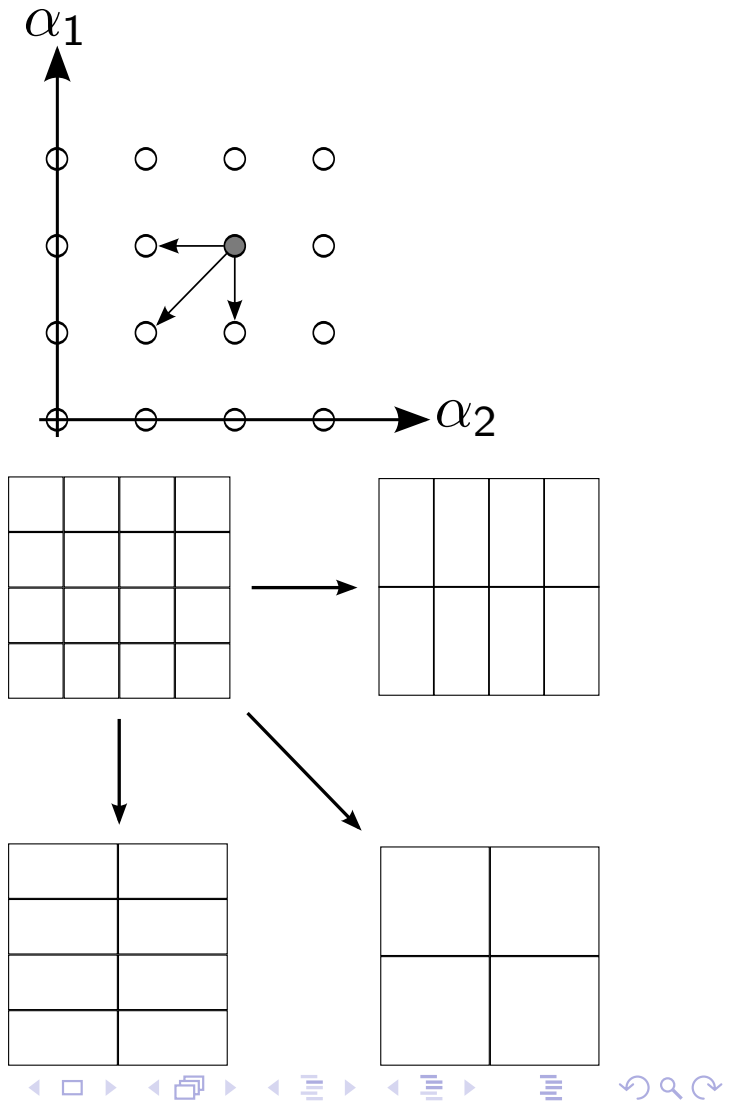
$$\Delta g_{\boldsymbol{\alpha}} = \left(\otimes_{i=1}^{d} \Delta_i\right) g_{\boldsymbol{\alpha}} = \sum_{\mathbf{j}\in\{0,1\}^d} (-1)^{|\mathbf{j}|} g_{\boldsymbol{\alpha}-\mathbf{j}}$$

with $|\mathbf{j}| = \sum_{i=1}^{d} j_i$. Requires 2^d evaluations of g on different grids.

Example: Computing g_α in $d = 2$

For $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$, we have

$$\begin{aligned}\Delta g_{(\alpha_1,\alpha_2)} &= \Delta_2(\Delta_1 g_{(\alpha_1,\alpha_2)}) \\ &= \Delta_2 \left(g_{\alpha_1,\alpha_2} - g_{\alpha_1-1,\alpha_2}\right) \\ &= g_{\alpha_1,\alpha_2} - g_{\alpha_1-1,\alpha_2} \\ &\quad - g_{\alpha_1,\alpha_2-1} + g_{\alpha_1-1,\alpha_2-1}.\end{aligned}$$



MIMC Estimator

Then, assuming that

$$\mathrm{E}[g_{\boldsymbol{\alpha}}] \to \mathrm{E}[g] \quad \text{as} \quad \alpha_i \to \infty \ \text{ for all } i = 1, \ldots, d,$$

it is not difficult to see that

$$\mathrm{E}[g] = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} \mathrm{E}[\Delta g_{\boldsymbol{\alpha}}]$$

MIMC Estimator

Then, assuming that

$$\mathrm{E}[g_{\boldsymbol{\alpha}}] \to \mathrm{E}[g] \quad \text{as} \quad \alpha_i \to \infty \ \text{ for all } i = 1, \ldots, d,$$

it is not difficult to see that

$$\mathrm{E}[g] = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} \mathrm{E}[\Delta g_{\boldsymbol{\alpha}}] \approx \sum_{\boldsymbol{\alpha} \in \mathcal{I}} \mathrm{E}[\Delta g_{\boldsymbol{\alpha}}]$$

where $\mathcal{I} \subset \mathbb{N}^d$ is a *properly chosen* index set.

MIMC Estimator

Then, assuming that

$$\mathrm{E}[g_{\boldsymbol{\alpha}}] \to \mathrm{E}[g] \quad \text{as} \quad \alpha_i \to \infty \ \text{ for all } i = 1, \ldots, d,$$

it is not difficult to see that

$$\mathrm{E}[g] = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} \mathrm{E}[\Delta g_{\boldsymbol{\alpha}}] \approx \sum_{\boldsymbol{\alpha} \in \mathcal{I}} \mathrm{E}[\Delta g_{\boldsymbol{\alpha}}]$$

where $\mathcal{I} \subset \mathbb{N}^d$ is a *properly chosen* index set.

As in MLMC, appoximating each term by independent MC samplers, the MIMC estimator can be written as

$$\mathcal{A}_{\mathsf{MIMC}}[g; \mathcal{I}] = \sum_{\boldsymbol{\alpha} \in \mathcal{I}} \frac{1}{M_{\boldsymbol{\alpha}}} \sum_{m=1}^{M_\alpha} \Delta g_{\boldsymbol{\alpha}}(\omega_{\boldsymbol{\alpha},m})$$

with *properly chosen* sample sizes $(M_{\boldsymbol{\alpha}})_{\boldsymbol{\alpha} \in \mathcal{I}}$.

Assumptions for MIMC

For every $\boldsymbol{\alpha}$, we assume the following

Assumption 1 (Bias) : $$E_{\boldsymbol{\alpha}} = |\mathrm{E}[\Delta g_{\boldsymbol{\alpha}}]| \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i w_i}$$

Assumption 2 (Variance) : $$V_{\boldsymbol{\alpha}} = \mathrm{Var}[\Delta g_{\boldsymbol{\alpha}}] \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i s_i},$$

Assumption 3 (Work) : $$W_{\boldsymbol{\alpha}} = \mathrm{Work}(\Delta g_{\boldsymbol{\alpha}}) \propto \prod_{i=1}^{d} \beta_i^{\alpha_i \gamma_i},$$

For positive constants $\gamma_i, w_i, s_i < 2w_i$ and for $i = 1 \dots d$.

$$\mathrm{Work}(\mathrm{MIMC}) = \sum_{\boldsymbol{\alpha} \in \mathcal{I}} M_{\boldsymbol{\alpha}} W_{\boldsymbol{\alpha}} \propto \sum_{\boldsymbol{\alpha} \in \mathcal{I}} M_{\boldsymbol{\alpha}} \left(\prod_{i=1}^{d} \beta_i^{\alpha_i \gamma_i} \right).$$

Remark on product rates

- The **Assumptions 1 & 2** in MIMC that the rates for each $\boldsymbol{\alpha}$ are *products of 1D rates*

$$E_{\boldsymbol{\alpha}} \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i w_i}, \qquad V_{\boldsymbol{\alpha}} \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i s_i}$$

  are stronger than the corresponding assumptions in MLMC.

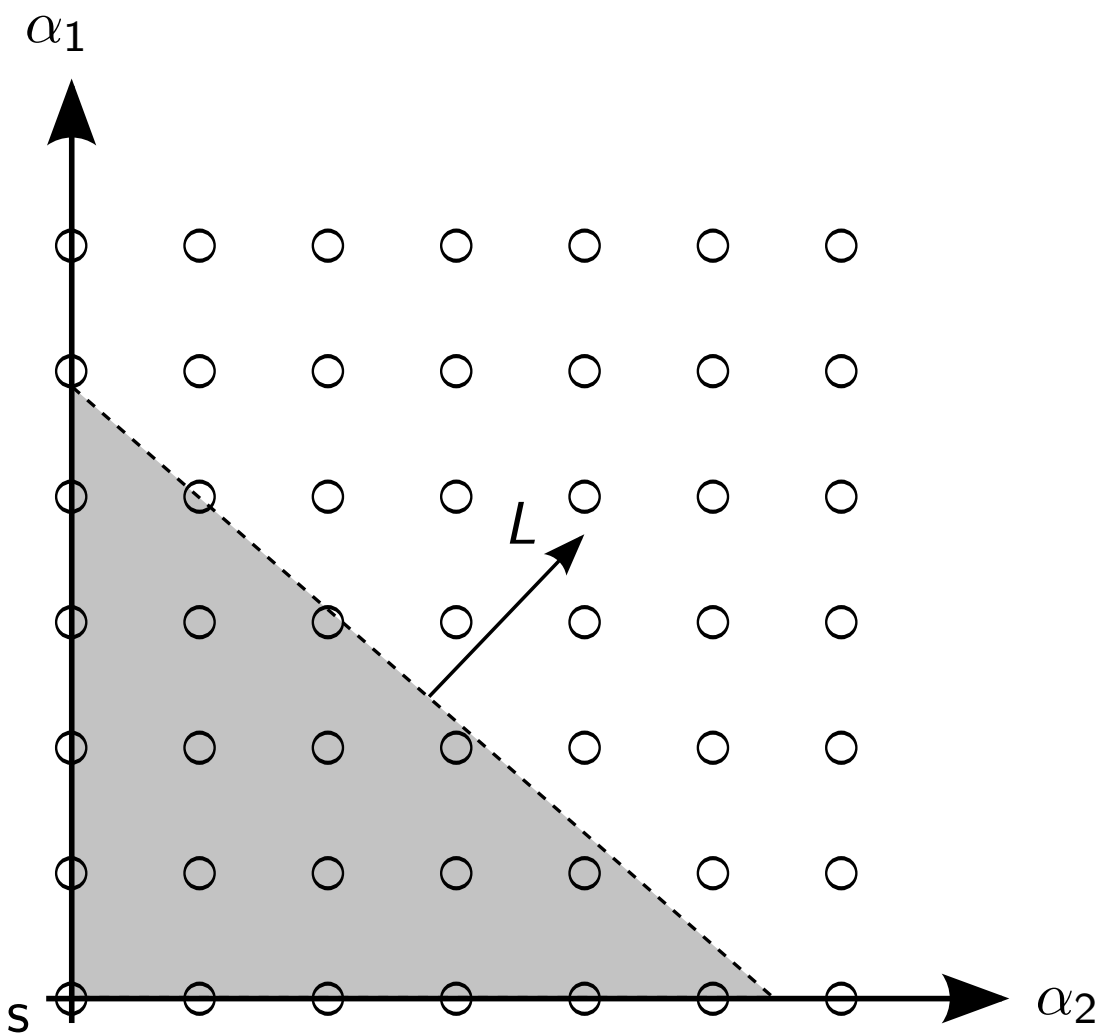
Remark on product rates

- The **Assumptions 1 & 2** in MIMC that the rates for each $\boldsymbol{\alpha}$ are *products of 1D rates*

$$E_{\boldsymbol{\alpha}} \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i w_i}, \qquad V_{\boldsymbol{\alpha}} \propto \prod_{i=1}^{d} \beta_i^{-\alpha_i s_i}$$

  are stronger than the corresponding assumptions in MLMC.
- They imply existence of mixed derivatives of the solution of the PDE (and possibly the solution of the adjoint problem associated to the functional Ψ), as opposed to standard derivatives for MLMC.

Optimal index set



Or find it adaptively!

Fully Isotropic Case: Rough noise case

Assume $w_i = w$, $s_i = s < 2w$, $\beta_i = \beta$ and $\gamma_i = \gamma$ for all $i \in \{1 \cdots d\}$. Then the optimal work is

$$\text{Work(MC)} = \mathcal{O}\left(\text{TOL}^{-2-\frac{d\gamma}{w}}\right).$$

$$\text{Work(MLMC)} = \begin{cases} \mathcal{O}\left(\text{TOL}^{-2}\right), & s > d\gamma, \\ \mathcal{O}\left(\text{TOL}^{-2}\left(\log\left(\text{TOL}^{-1}\right)\right)^2\right), & s = d\gamma, \\ \mathcal{O}\left(\text{TOL}^{-\left(2+\frac{(d\gamma - s)}{w}\right)}\right), & s < d\gamma. \end{cases}$$

$$\text{Work(MIMC)} = \begin{cases} \mathcal{O}\left(\text{TOL}^{-2}\right), & s > \gamma, \\ \mathcal{O}\left(\text{TOL}^{-2}\left(\log\left(\text{TOL}^{-1}\right)\right)^{2d}\right), & s = \gamma, \\ \mathcal{O}\left(\text{TOL}^{-\left(2+\frac{\gamma - s}{w}\right)}\log\left(\text{TOL}^{-1}\right)^{(d-1)\frac{\gamma - s}{w}}\right), & s < \gamma. \end{cases}$$

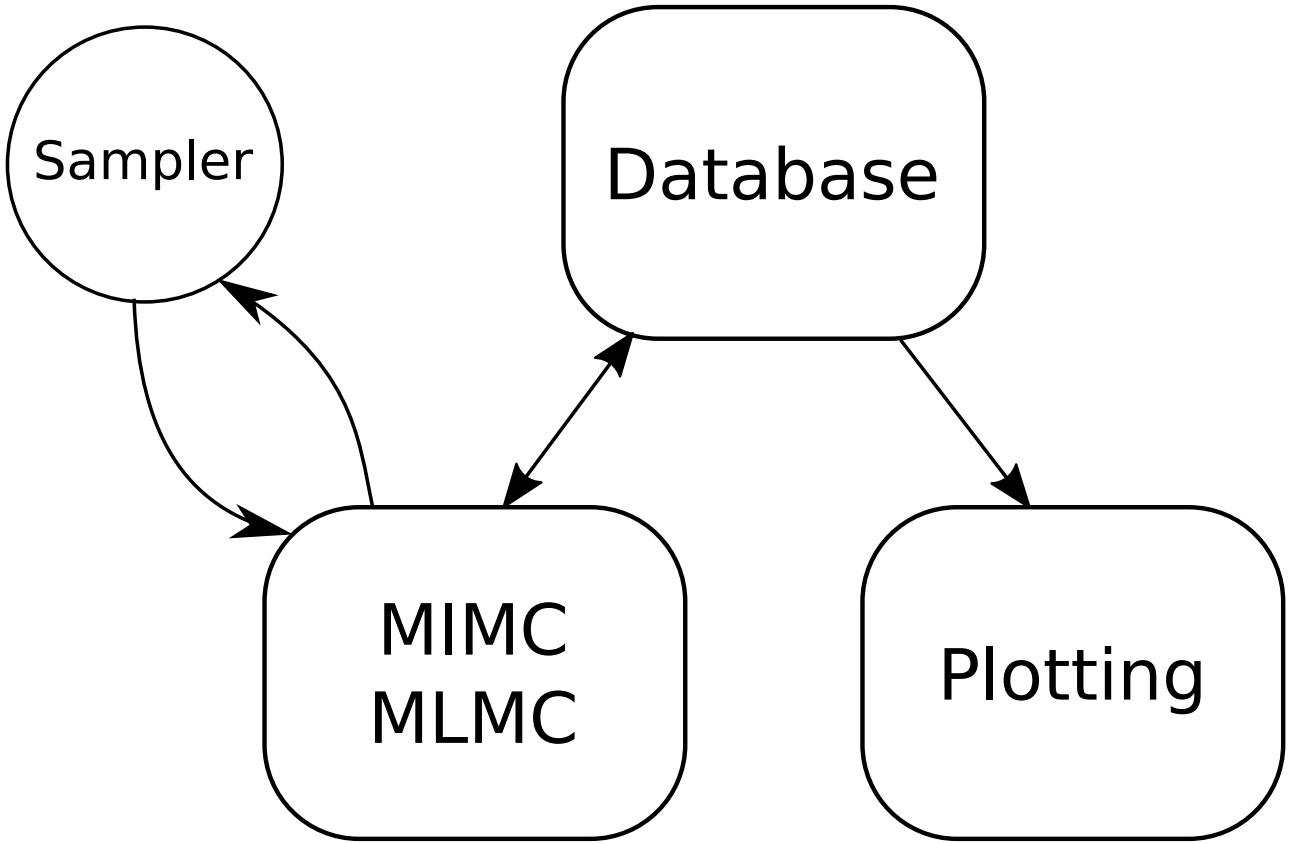
- These methods and others are common in UQ applications.
- UQ research requires systematic studies of deterministic or random runs.
- `mimclib` is a library that was developed with that objective in mind.
- `mimclib` can also be used for non-academic purpose too.

Vision

- Provide an **easy to use**, **customizable** and **extendable** open source library for UQ problems, both forward and inverse.
- Multilevel and Multi-index versions of Monte Carlo, Quasi Monte Carlo, Stochastic collocation, Least square projection among others.
- Support parallel computation whenever possible.
- Provide easy to use storage facility.
- Provide easy to customize plotting facility (for common plots).
- Provide some simple (and not so simple) test cases.
- Use `Python` for easier implementation of most parts of code and use object code (`C++` or `FORTRAN`) for computationally expensive parts.

What has been done, mimclib 0.2.0.dev0

- Multilevel and Multi-index versions of Monte Carlo, Stochastic Collocation and Random Projection. Adaptive and non-adaptive construction of index sets.
- Provide easy to use storage facility in SQLite and MySQL.
- Provide easy to customize plotting facility using matplotlib (for common plots).
- Provide easy to run test cases.
- Documentation is still in progress (these slides are part of it).
- MPI friendly, but parallelization has to be implemented by user.
- Interface is written with the other features in mind.


Sampler
Database
MIMC
MLMC
Plotting

Why?

- Python
  - Open source. No need for licensing
  - An easy to use programming language. Familiar to MATLAB users (Especially with numpy and matplotlib)
  - Can call object code for computationally expensive parts, e.g., index set constructors and samplers.
- Database engine: MySQL and SQLite
  - Relatively easy data modifications and querying.
  - Allows asynchronous access which is ideal for parallel computation.
  - Allows remote access.
  - Optimal storage and data querying.
  - No need to worry about it for average user.

Installation

- Prerequisites: `gcc` (supporting `c++11`), `python2.7`, `pip`.
- First step:

```
> git clone \
    https://github.com/StochasticNumerics/mimclib.git
> cd mimclib
> make pip
```

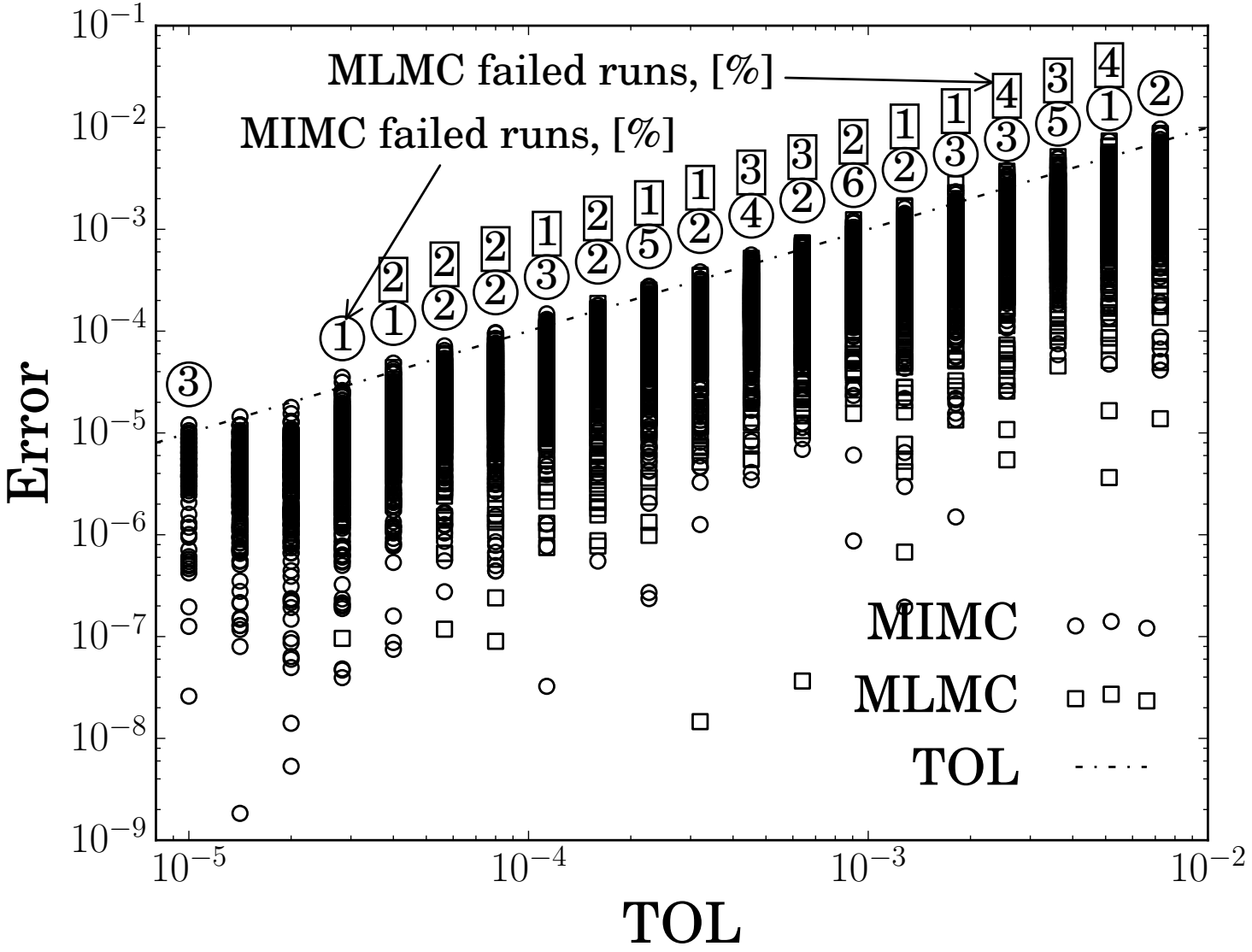
  Or just google: "mimclib github"
- Done!.
- A bit more complicated to install `MySQL`. Not for the faint of heart but very convenient if you are planning to run thousands of simulations on a cluster.

Minimal Sampler

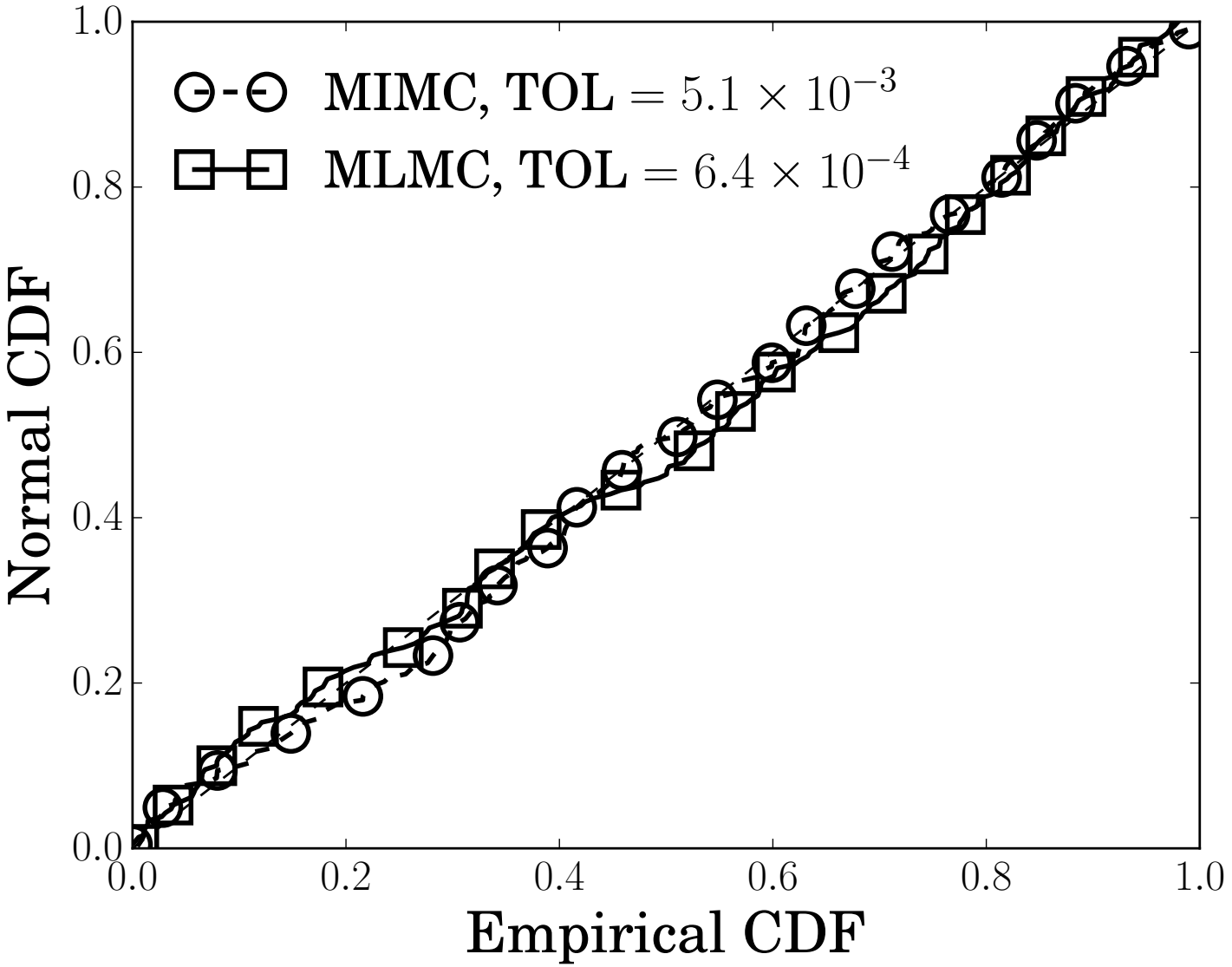
```
def mySampleQoI(run, inds, M):
    #  run: is a reference to current run
    # inds: is a dxN array that specifies the
    #         discretization-level along each dimension
    #    M: number of samples

    return solves, t
    # solves: and NxM array that returns the samples
    #           for each discretization
    #      t: time taken to generate these samples.
```

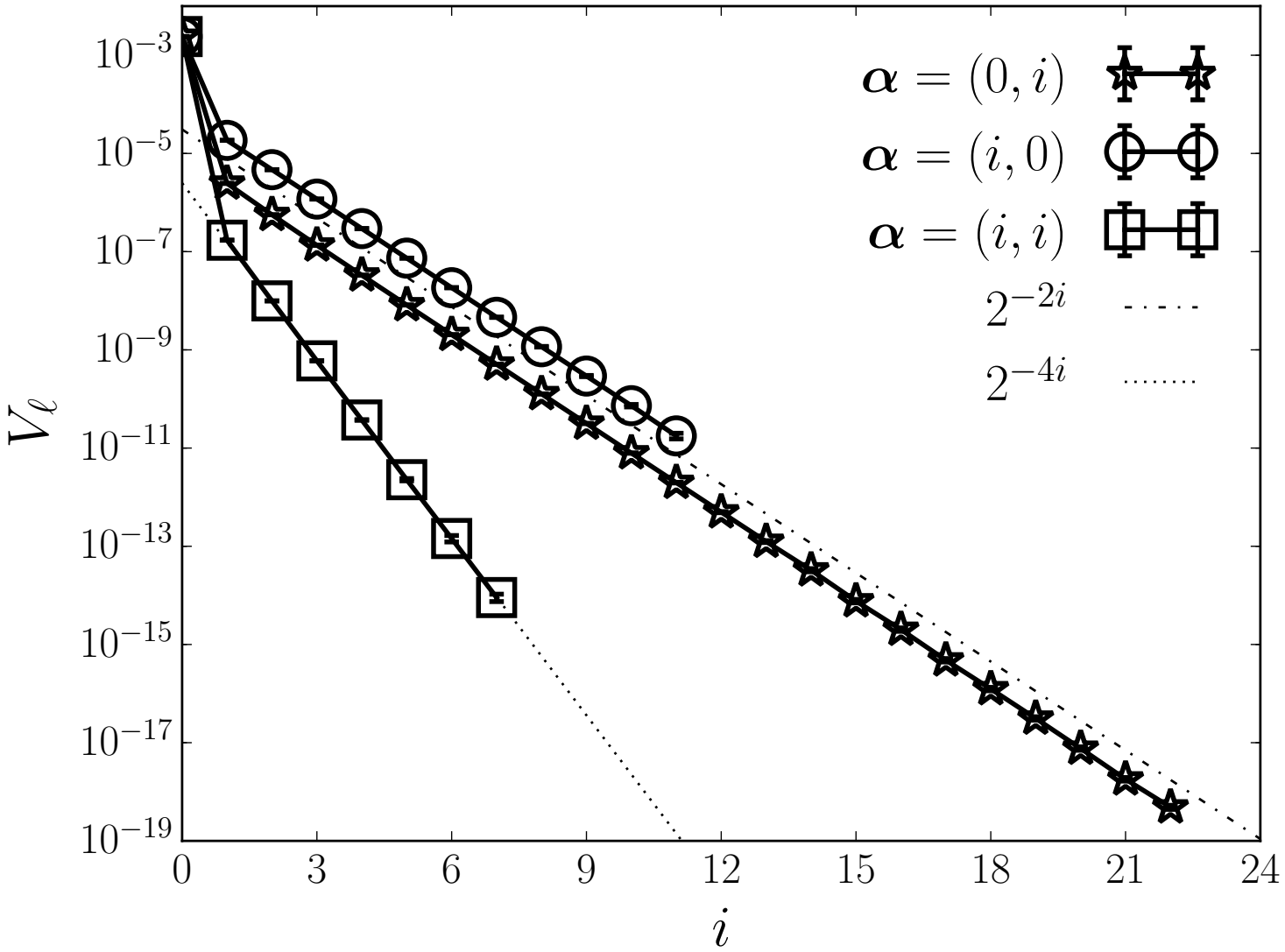
Errors, customised



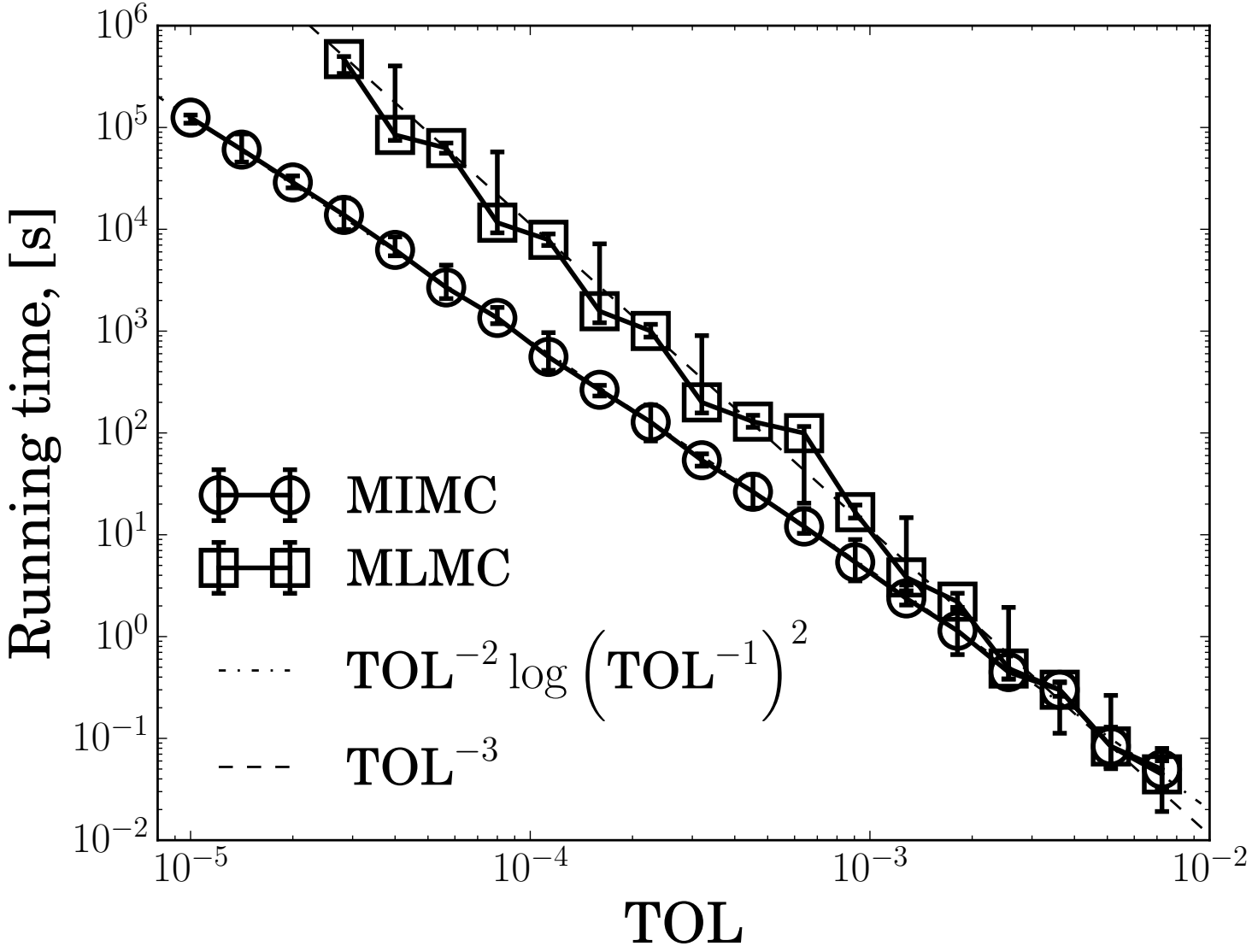
PP-plot


MIMC, TOL $= 5.1 \times 10^{-3}$
MLMC, TOL $= 6.4 \times 10^{-4}$
Normal CDF
Empirical CDF
0.0
0.2
0.4
0.6
0.8
1.0

Variance convergence



Running time



Conclusions

- Monte Carlo methods have slow convergence but do not suffer from curse-of-dimensionality (in parameter space).
- MLMC allows tackling problems that require expensive computation but, like Monte Carlo, suffers from curse-of-dimensionality (in physical space).
- MIMC exploits extra regularity to get a complexity that is independent of (or slowly growing with) number of physical dimensions.

Conclusions

- Monte Carlo methods have slow convergence but do not suffer from curse-of-dimensionality (in parameter space).
- MLMC allows tackling problems that require expensive computation but, like Monte Carlo, suffers from curse-of-dimensionality (in physical space).
- MIMC exploits extra regularity to get a complexity that is independent of (or slowly growing with) number of physical dimensions.
- Research in UQ methods is booming. `mimclib` can provide you with a base to start from.
- `mimclib` is still growing, rapidly, to adapt to different applications and different samplers.
- If you feel that `mimclib` is missing a feature, fork it!

mimclib needs your help!

Questions?