Multilevel and Multi-Index Monte Carlo: Theory, Practice, and `mimclib`

Abdul-Lateef Haji-Ali

School of Mathematical and Computer Sciences, Heriot-Watt University

Outline

1. Theory: Monte Carlo methods
   - Problem
   - Monte Carlo
   - Multilevel Monte Carlo (MLMC)
   - Multi-Index Monte Carlo (MIMC)

2. Implementation: `mimclib`
   - Library Overview
   - Installation
   - Example output from MIMC example

The central question

Given an $\mathbb{R}^n$-valued random variable, X, a function, $g : \mathbb{R}^n \to \mathbb{R}$, assume that we are interested in computing the quantity

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

Challenges:

- Only approximate samples of X can be generated, e.g., X is the solution of an SDE.
- g cannot be computed exactly or cheaply, e.g., involves FEM approximation, or a nested expectation.

The central question

Given an $\mathbb{R}^n$-valued random variable, X, a function, $g : \mathbb{R}^n \to \mathbb{R}$, assume that we are interested in computing the quantity

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

Challenges:

- Only approximate samples of X can be generated, e.g., X is the solution of an SDE.
- g cannot be computed exactly or cheaply, e.g., involves FEM approximation, or a nested expectation.

Setup

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

$$P(|\mathcal{A} - \mathrm{E}[\, g(X)\,]| \leq \varepsilon) \ \geq \eta$$

for a given accuracy ε and a given confidence level determined by $0 < \eta < 1$.

Instead, we impose the following, more restrictive, two constraints:

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

$$\textbf{Statistical constraint:} \qquad \mathrm{Var}[\, \mathcal{A}\,] \leq \theta^2\, \varepsilon^2$$

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

Setup

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

$$P(|\mathcal{A} - \mathrm{E}[\, g(X)\,]| \leq \varepsilon) \ \geq \eta$$

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

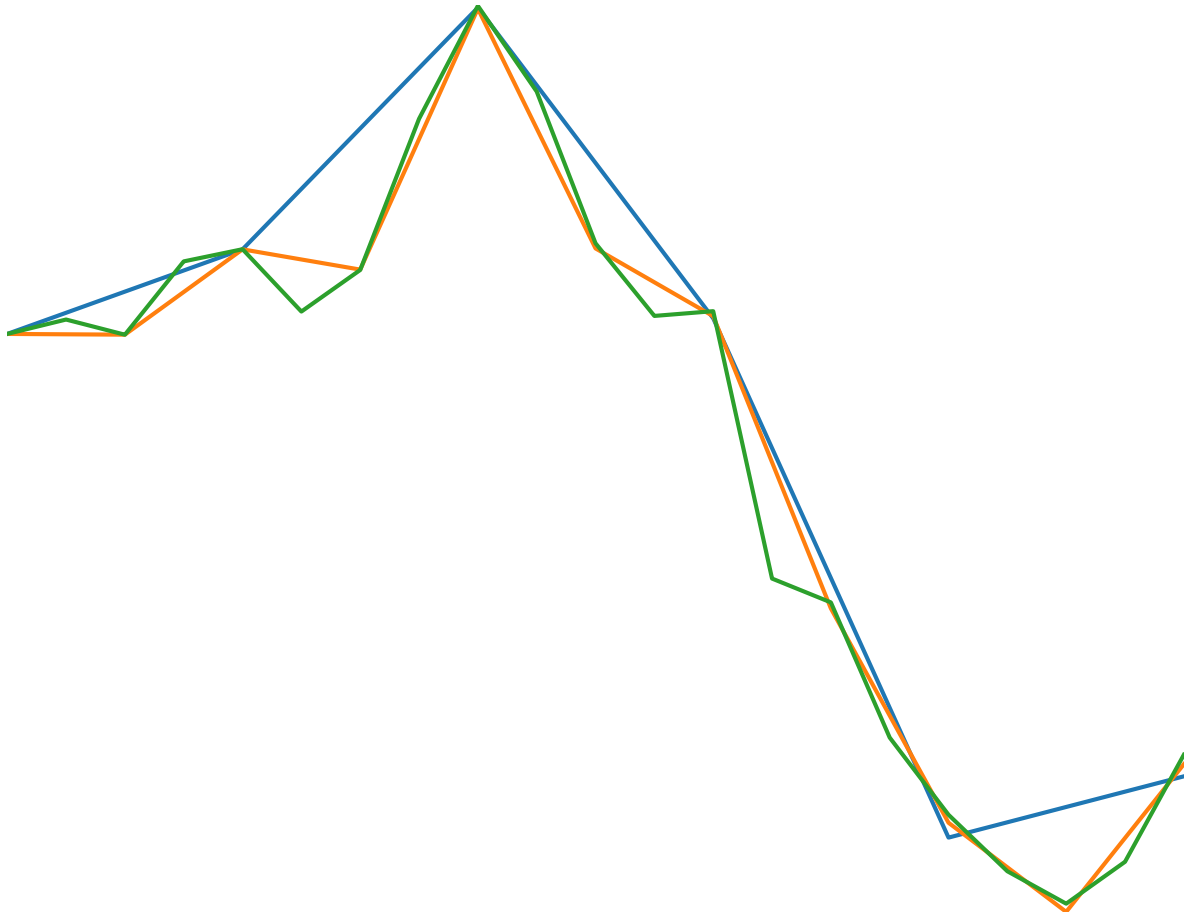
$$\textbf{Bias constraint:} \qquad |\mathrm{E}[\, \mathcal{A} - g(X)\,]| \leq (1-\theta)\varepsilon,$$

$$\textbf{Statistical constraint:} \qquad \mathrm{Var}[\, \mathcal{A}\,] \leq \theta^2\, \varepsilon^2$$

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

Numerical approximation

Notation: g_ℓ for $\ell \in \mathbb{N}$ is the approximation of g calculated based on some discretization. As ℓ increases, both the accuracy and cost per sample increase.



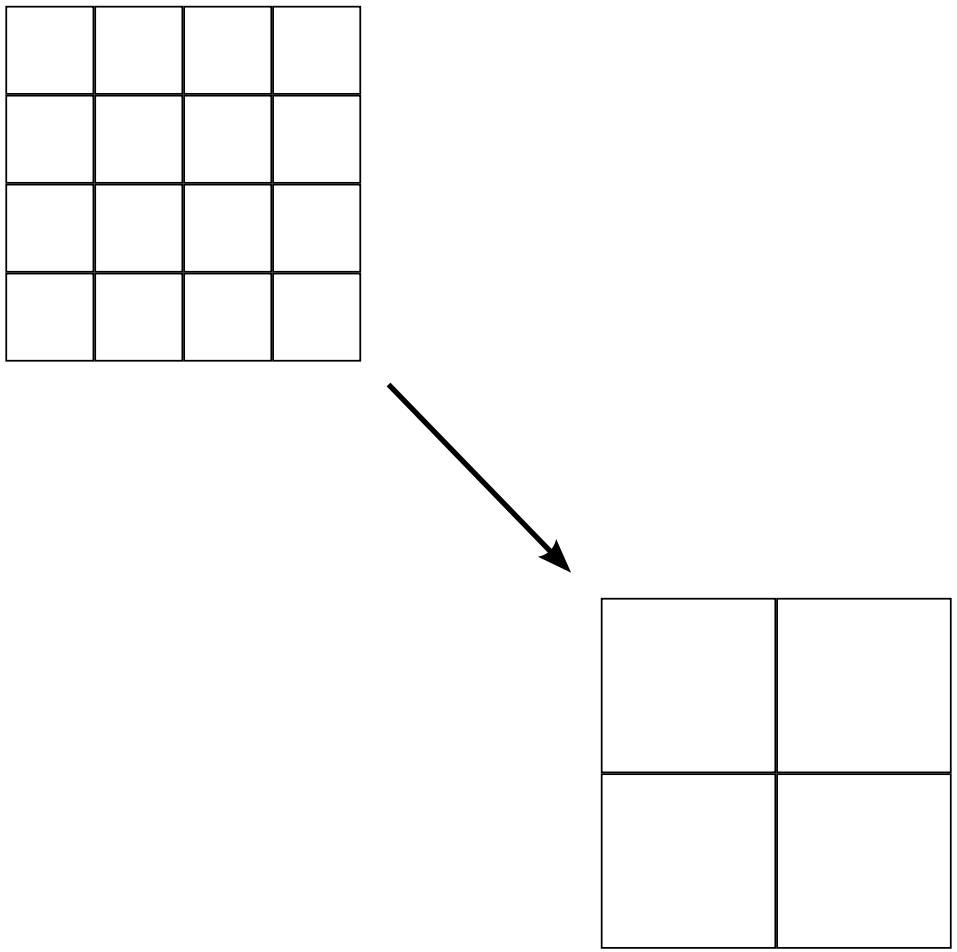
Monte Carlo

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

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

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

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

Main idea: variance reduction using cheaper approximations.

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

Observe the telescopic identity

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

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

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

Main idea: variance reduction using cheaper approximations.

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

Observe the telescopic identity

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

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

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

Main idea: variance reduction using cheaper approximations.

Assumptions

Assumption MC1 (Bias): $E_\ell := |\mathrm{E}[\, g - g_\ell \,]| \lesssim 2^{-\alpha \ell}$,
Assumption MC2 (Work): $\mathrm{Work}[\Delta_\ell g] \lesssim 2^{d\gamma\ell}$,
Assumption MLMC3 (Variance): $V_\ell := \mathrm{Var}[\, \Delta_\ell g \,] \lesssim 2^{-\beta\ell}$

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

Recall: Optimal cost of Monte Carlo is $\mathcal{O}(\varepsilon^{-2-\frac{d\gamma}{\alpha}})$

The optimal cost of MLMC is

$$\begin{cases} \mathcal{O}(\varepsilon^{-2}) & \beta > d\gamma \\ \mathcal{O}(\varepsilon^{-2}) \left(\log\left(\varepsilon^{-1}\right)\right)^2 & \beta = d\gamma \\ \mathcal{O}(\varepsilon^{-2-\frac{d\gamma-\beta}{\alpha}}) & \beta < d\gamma \end{cases}$$

Notice: the cost increases with increasing work (exponentially in d).

Assumptions

Assumption MC1 (Bias): $E_\ell := |\mathrm{E}[\, g - g_\ell \,]| \lesssim 2^{-\alpha \ell}$,
Assumption MC2 (Work): $\mathrm{Work}[\Delta_\ell g] \lesssim 2^{d\gamma\ell}$,
Assumption MLMC3 (Variance): $V_\ell := \mathrm{Var}[\, \Delta_\ell g \,] \lesssim 2^{-\beta \ell}$

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

Recall: Optimal cost of Monte Carlo is $\mathcal{O}(\varepsilon^{-2-\frac{d\gamma}{\alpha}})$

The optimal cost of MLMC is

$$\begin{cases} \mathcal{O}(\varepsilon^{-2}) & \beta > d\gamma \\ \mathcal{O}(\varepsilon^{-2}) \left(\log\left(\varepsilon^{-1}\right)\right)^2 & \beta = d\gamma \\ \mathcal{O}(\varepsilon^{-2-\frac{d\gamma-\beta}{\alpha}}) & \beta < d\gamma \end{cases}$$

Notice: the cost increases with increasing work (exponentially in d).

Assumptions

Assumption MC1 (Bias): $E_\ell := |\mathrm{E}[\, g - g_\ell \,]| \lesssim 2^{-\alpha \ell}$,
Assumption MC2 (Work): $\mathrm{Work}[\Delta_\ell g] \lesssim 2^{d\gamma\ell}$,
Assumption MLMC3 (Variance): $V_\ell := \mathrm{Var}[\, \Delta_\ell g \,] \lesssim 2^{-\beta \ell}$

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

Recall: Optimal cost of Monte Carlo is $\mathcal{O}(\varepsilon^{-2-\frac{d\gamma}{\alpha}})$

The optimal cost of MLMC is

$$\begin{cases} \mathcal{O}(\varepsilon^{-2}) & \beta > d\gamma \\ \mathcal{O}(\varepsilon^{-2}) \left(\log\left(\varepsilon^{-1}\right)\right)^2 & \beta = d\gamma \\ \mathcal{O}(\varepsilon^{-2-\frac{d\gamma-\beta}{\alpha}}) & \beta < d\gamma \end{cases}$$

Notice: the cost increases with increasing work (exponentially in d).

In practice: Continuation MLMC

- To compute M_ℓ, we need to find estimates of $V_\ell := \mathrm{Var}[\,\Delta_\ell \mathit{func}\,]$.
- To determine L, we need reliable estimate of E_ℓ.
- Instead of running for the small required ε, CMLMC runs a sequence of MLMC realisations, for decreasing tolerances, ending with the required ε.
- 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.
- This works as long as the Kurtosis does not increase too much with ℓ.
- 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.
- All implemented in `mimclib`, see later.

In practice: Continuation MLMC

- To compute M_ℓ, we need to find estimates of $V_\ell := \mathrm{Var}[\,\Delta_\ell \mathit{func}\,]$.
- To determine L, we need reliable estimate of E_ℓ.
- Instead of running for the small required ε, CMLMC runs a sequence of MLMC realisations, for decreasing tolerances, ending with the required ε.
- 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.
- This works as long as the Kurtosis does not increase too much with ℓ.
- 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.
- All implemented in `mimclib`, see later.

A simple problem

- Consider this problem

$$\mathrm{E}[\, g(\mathrm{E}[\, f(X_T) \,|\, X_H \,]) \,]$$

for an SDE solution,

$$\mathrm{d}X_t = \mu_t \,\mathrm{d}t + \sigma_t \,\mathrm{d}W_t$$

- How do we approximate it?
  - Requires approximation of outer process X_H
  - Requires approximation of inner expectation $\mathrm{E}[\, X_T \,|\, X_H \,]$
  - Requires approximation of inner process X_T, given X_H.
- Using $2^{-\ell}$ for the step-size of the process and 2^{ℓ} inner samples the cost of approximating a single sample is $\mathcal{O}(2^{2\ell})$.
- The variance of $\Delta_\ell g$, given these approximations, converges slower than $\mathcal{O}(2^{-2\ell})$, so complexity is not optimal.
- Similar problem if particle system is used.

A simple problem

- Consider this problem

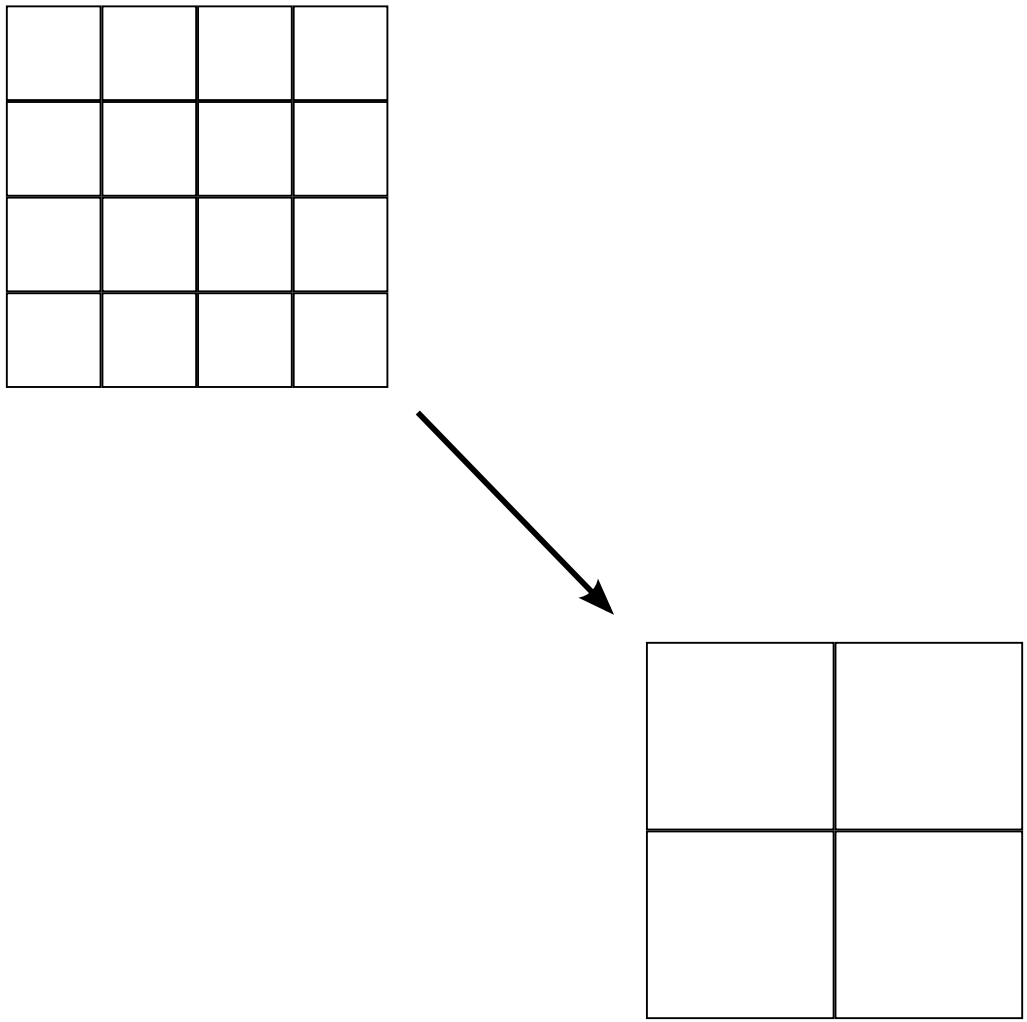
$$\mathrm{E}[\, g(\mathrm{E}[\, f(X_T) \,|\, X_H \,]) \,]$$

  for an SDE solution,

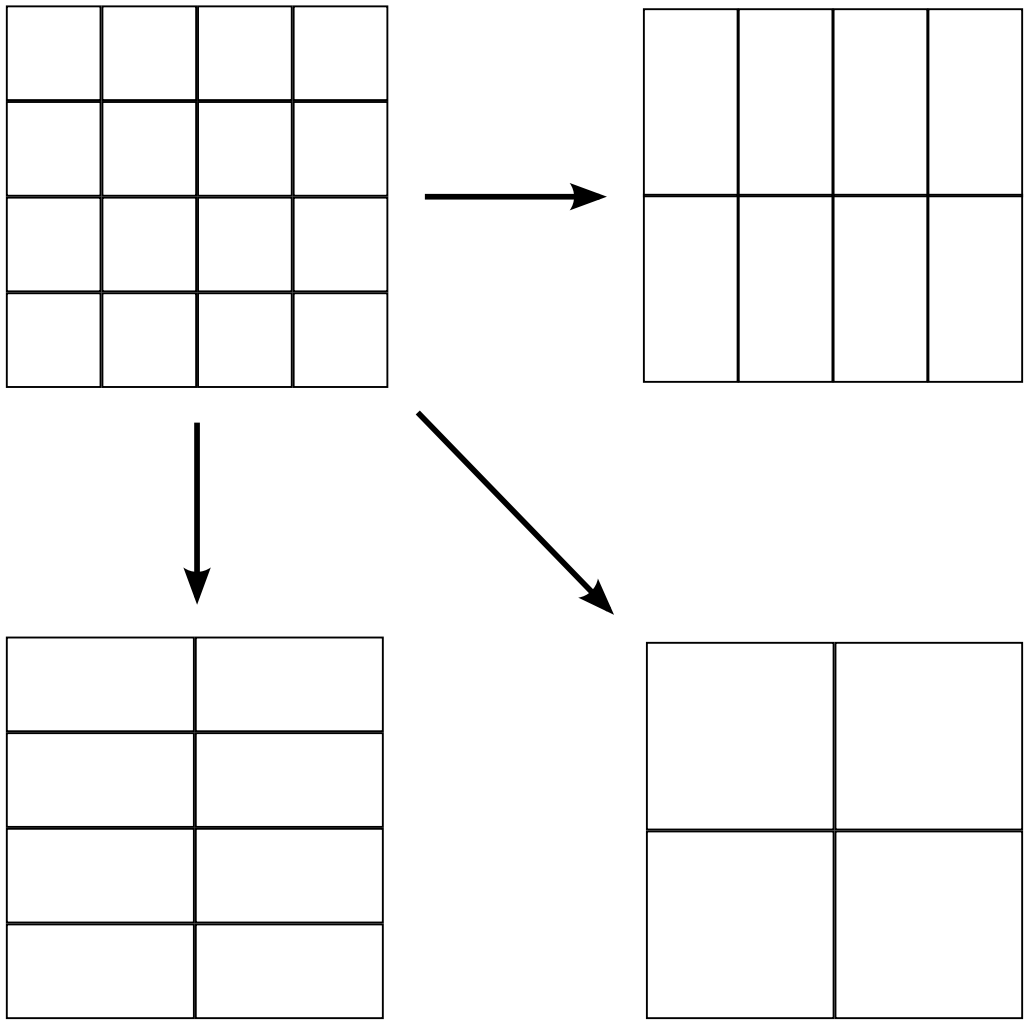
$$\mathrm{d}X_t = \mu_t \,\mathrm{d}t + \sigma_t \,\mathrm{d}W_t$$

- How do we approximate it?
  - Requires approximation of outer process X_H
  - Requires approximation of inner expectation $\mathrm{E}[\, X_T \,|\, X_H \,]$
  - Requires approximation of inner process X_T, given X_H.
- Using $2^{-\ell}$ for the step-size of the process and 2^{ℓ} inner samples the cost of approximating a single sample is $\mathcal{O}(2^{2\ell})$.
- The variance of $\Delta_\ell g$, given these approximations, converges slower than $\mathcal{O}(2^{-2\ell})$, so complexity is not optimal.
- Similar problem if particle system is used.

Variance reduction: MLMC



Variance reduction: Further potential



MIMC Estimator (Haji-Ali et al., 2017)

Consider discretization parameter, $\ell = (\ell_1, \ell_2, \ldots, \ell_d) \in \mathbb{N}^d$, and use the same notation g_ℓ the approximation of g calculated using a discretization defined by ℓ, e.g., using 2^{ℓ_1} time-steps and 2^{ℓ_2} inner samples.

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

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

Construct the first order mixed difference

$$\begin{aligned} \Delta_\ell g &= \left(\otimes_{i=1}^d \Delta_i \right) g_\ell \\ &= g_{\ell_1,\ell_2} - g_{\ell_1-1,\ell_2} - g_{\ell_1,\ell_2-1} + g_{\ell_1-1,\ell_2-1} \end{aligned}$$

Requires 2^d evaluations of g with different discretizations.

MIMC Estimator (Haji-Ali et al., 2017)

Consider discretization parameter, $\ell = (\ell_1, \ell_2, \ldots, \ell_d) \in \mathbb{N}^d$, and use the same notation g_ℓ the approximation of g calculated using a discretization defined by ℓ, e.g., using 2^{ℓ_1} time-steps and 2^{ℓ_2} inner samples.

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

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

Construct the first order mixed difference

$$\begin{aligned} \Delta_\ell g &= \left(\otimes_{i=1}^d \Delta_i \right) g_\ell \\ &= g_{\ell_1,\ell_2} - g_{\ell_1-1,\ell_2} - g_{\ell_1,\ell_2-1} + g_{\ell_1-1,\ell_2-1} \end{aligned}$$

Requires 2^d evaluations of g with different discretizations.

MIMC Estimator

Then, assuming that

$$\mathsf{E}[\,g_\ell\,] \to \mathsf{E}[\,g\,] \quad \text{as} \quad \ell_i \to \infty \ \text{ for all } i = 1, \ldots, d,$$

it is not difficult to see that

$$\mathsf{E}[\,g\,] = \sum_{\ell \in \mathbb{N}^d} \mathsf{E}[\,\Delta_\ell g\,] \approx \sum_{\ell \in \mathcal{I}} \mathsf{E}[\,\Delta_\ell g\,]$$

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

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

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

MIMC Estimator

Then, assuming that

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

it is not difficult to see that

$$\mathrm{E}[\,g\,] = \sum_{\ell \in \mathbb{N}^d} \mathrm{E}[\,\Delta_\ell g\,] \approx \sum_{\ell \in \mathcal{I}} \mathrm{E}[\,\Delta_\ell g\,]$$

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

$$\mathcal{A}_{\mathrm{MIMC}}[g; \mathcal{I}] = \sum_{\ell \in \mathcal{I}} \frac{1}{M_\ell} \sum_{m=1}^{M_\ell} \Delta_\ell g(\omega_{\ell,m})$$

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

MIMC Estimator

Then, assuming that

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

it is not difficult to see that

$$\mathrm{E}[\,g\,] = \sum_{\ell \in \mathbb{N}^d} \mathrm{E}[\,\Delta_\ell g\,] \approx \sum_{\ell \in \mathcal{I}} \mathrm{E}[\,\Delta_\ell g\,]$$

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

$$\mathcal{A}_{\mathrm{MIMC}}[g; \mathcal{I}] = \sum_{\ell \in \mathcal{I}} \frac{1}{M_\ell} \sum_{m=1}^{M_\ell} \Delta_\ell g(\omega_{\ell,m})$$

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

Assumptions for MIMC

For every ℓ, we assume the following

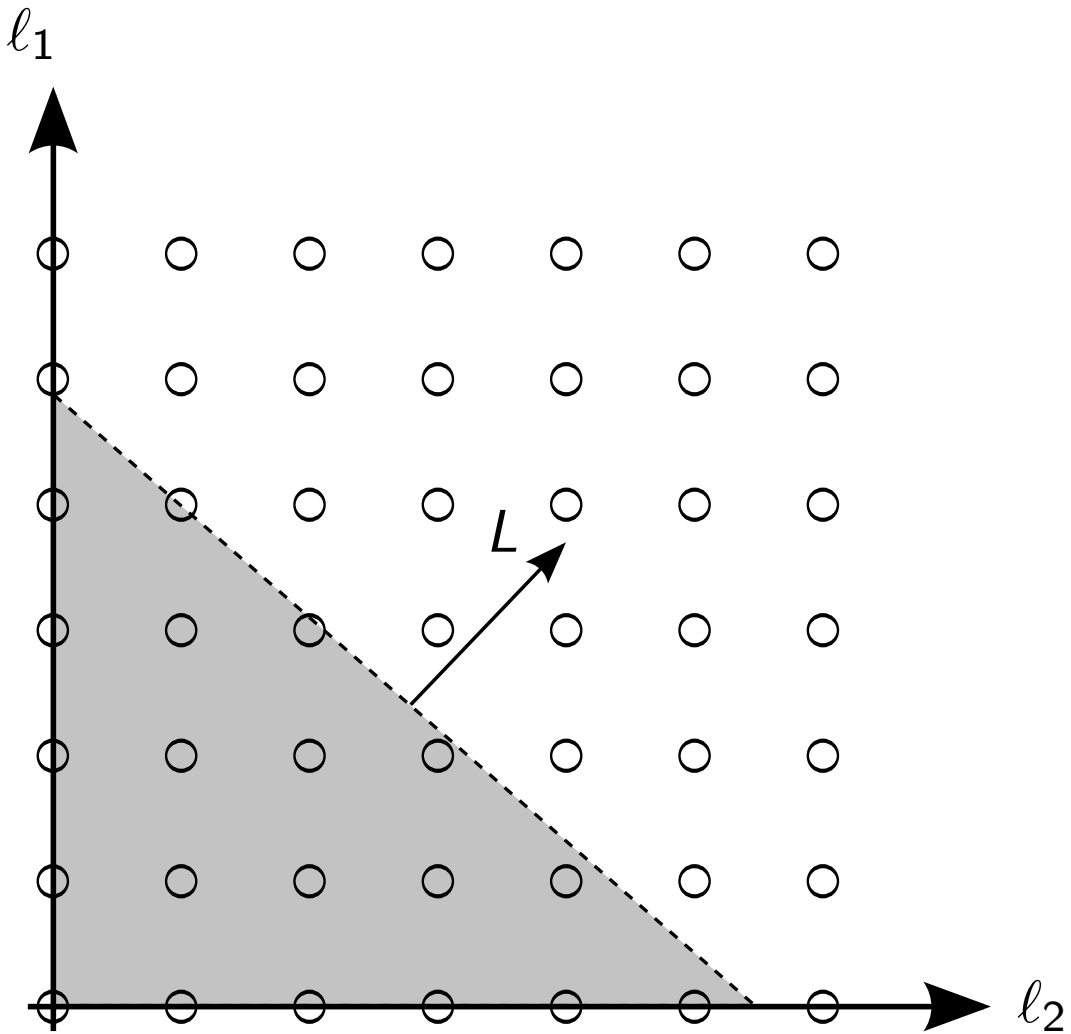
Assumption MIMC1 (Bias) : $$E_\ell = |\mathrm{E}[\,\Delta_\ell g\,]| \propto \prod_{i=1}^{d} 2^{-\alpha_i\,\ell_i}$$

Assumption MIMC2 (Work) : $$W_\ell = \mathrm{Work}(\Delta_\ell g) \propto \prod_{i=1}^{d} 2^{\gamma_i \ell_i},$$

Assumption MIMC3 (Variance) : $$V_\ell = \mathrm{Var}[\,\Delta_\ell g\,] \propto \prod_{i=1}^{d} 2^{-\beta_i\,\ell_i},$$

- The MIMC assumptions are stronger than the corresponding assumptions in MLMC.
  They imply existence of mixed derivatives of the underlying map (and possibly the solution of the adjoint problem associated to the functional g), as opposed to first-order derivatives for MLMC.

Optimal index set, $\mathcal{I}$



Or find it adaptively!

Fully Isotropic case

Assume $\alpha_i = \alpha$, $\beta_i = \beta < 2\alpha$ and $\gamma_i = \gamma$ for all $i \in \mathcal{I}$. Then the optimal work is

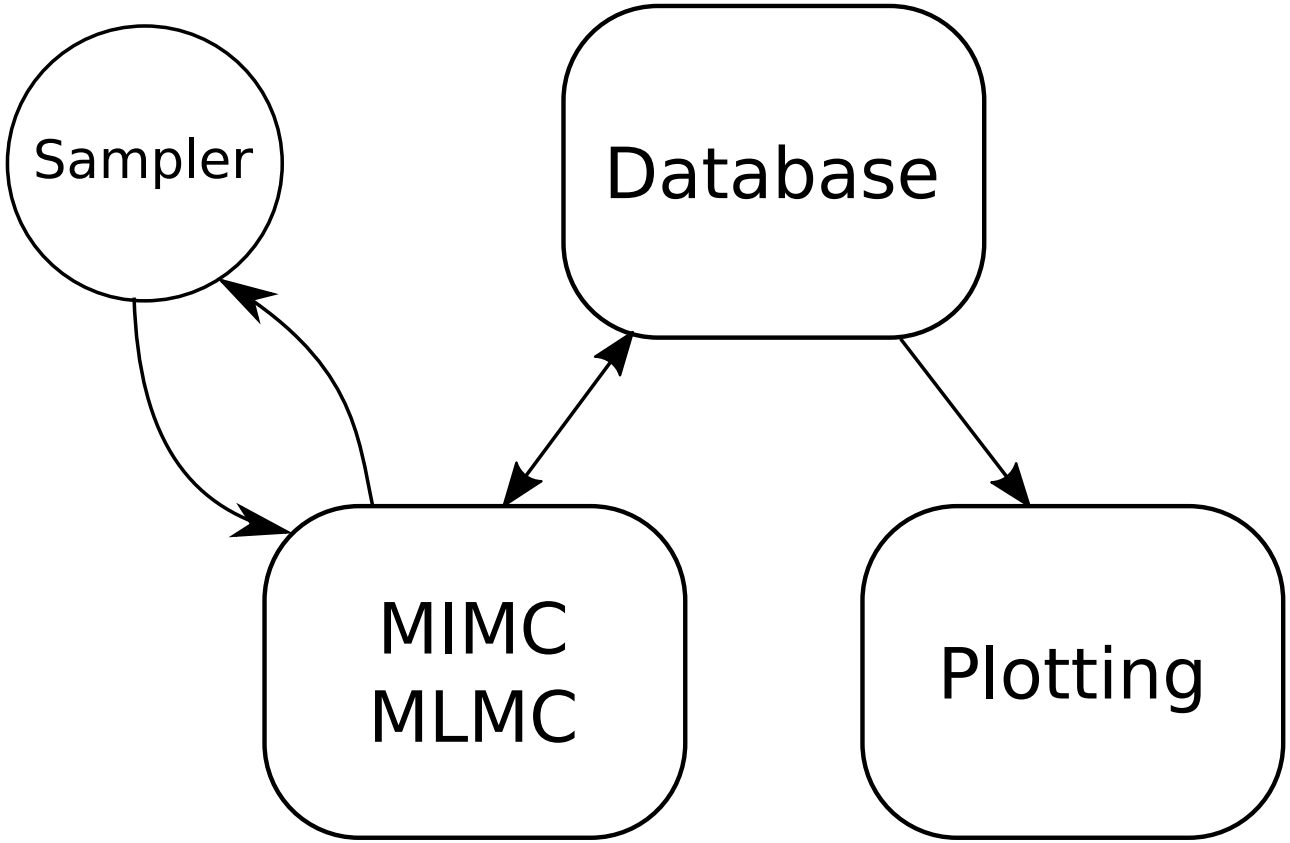
$$\text{Work(MC)} = \mathcal{O}(\varepsilon^{-2-\frac{d\gamma}{\alpha}}).$$

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

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

mimclib

- These methods and others are common in UQ applications.
- UQ research and applications require systematic studies of random runs.
- `mimclib` is a library that was developed with that objective in mind.
  - 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, Stochastic collocation, Least square projection.
  - Parallel computation-friendly.
  - Provide easy to use storage facility (`SQLite` and `MySQL`, allowing asynchronous, remote access).
  - Provide easy to customize plotting facility (for common plots) – using `matplotlib`.
  - Provide some simple (and not so simple) test cases.
  - Use `Python` for easier implementation of most parts of code and use object code (`C++`) for computationally expensive parts.
- `mimclib` can also be used for non-academic purpose too.


Sampler
Database
MIMC
MLMC
Plotting

Installation

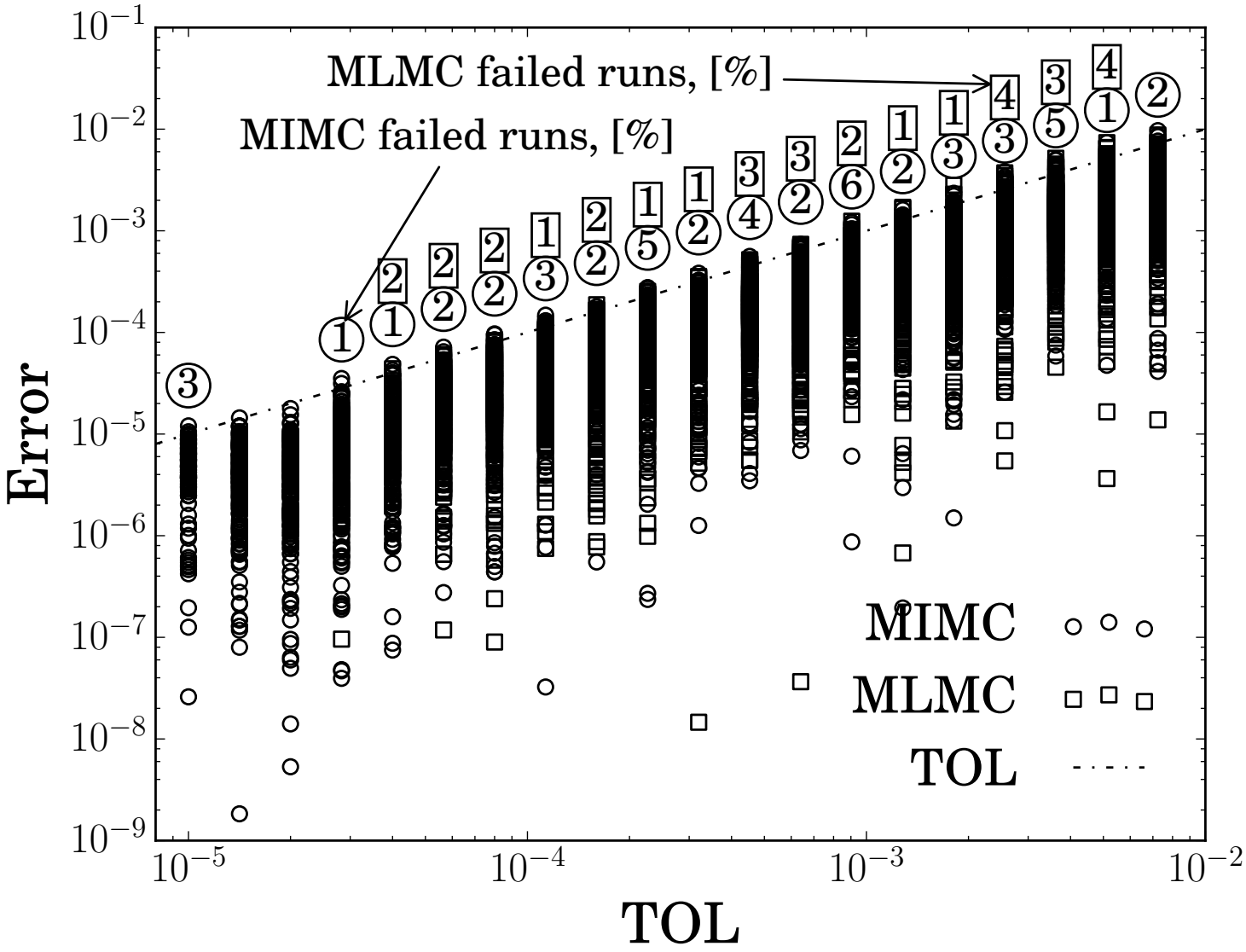
- Prerequisites: `gcc` (supporting `c++11`), `Python 3.4`, `pip`.
- ```
  > git clone \
      https://github.com/haji-ali/mimclib.git
  > cd mimclib
  > make pip
  ```
  Or just google: "mimclib github"
- Done!.
- Uses `sqlite3` by default. A bit more complicated to install `MySQL` but very convenient if you are planning to run thousands of simulations on a cluster.
  ```

Minimal Sampler

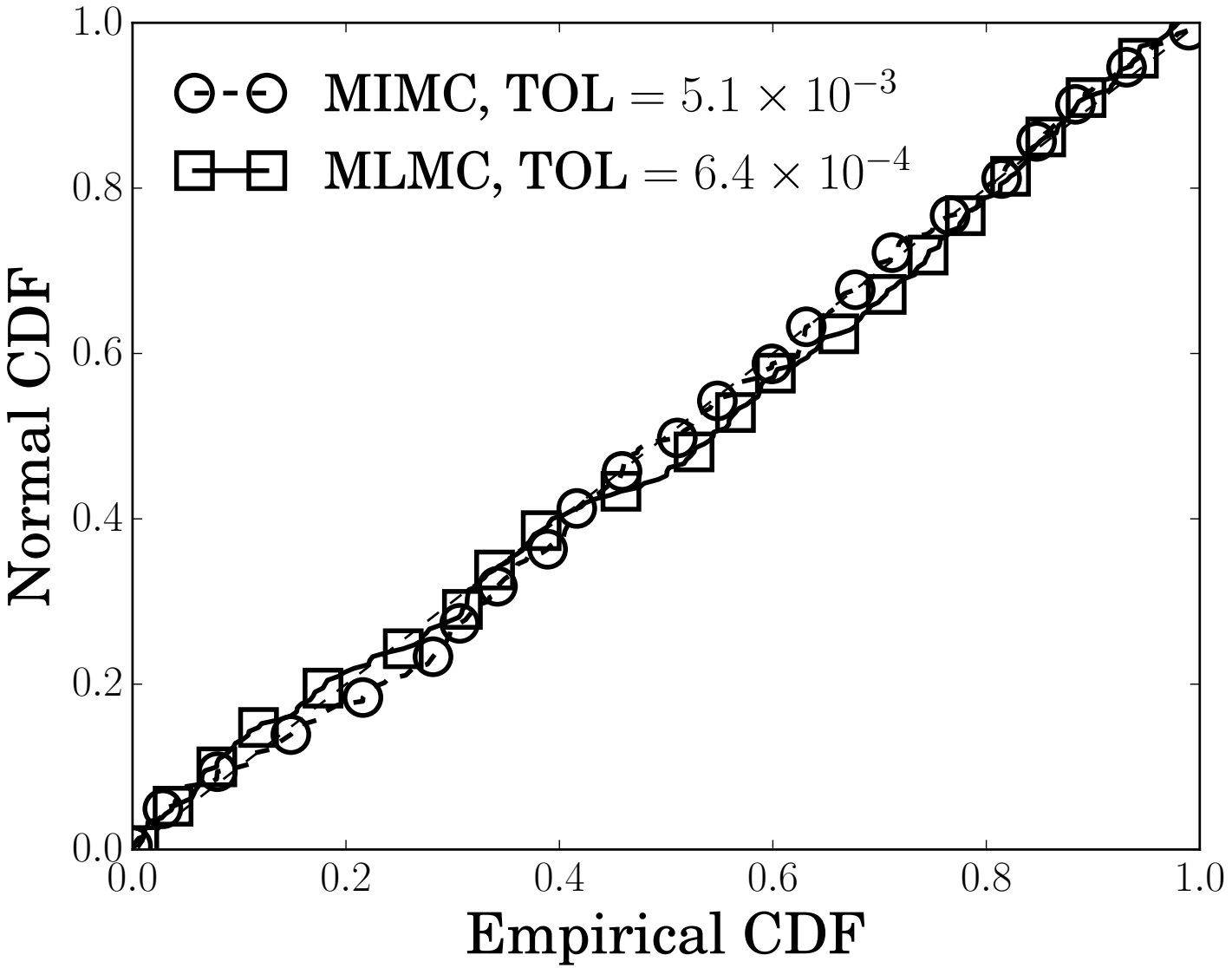
```
def mySampleQoI(run, inds, M):
    #  run: is a reference to current run
    # inds: is a dxN array that specifies the
    #         discretization-level along each dimension
    #    M: number of samples

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

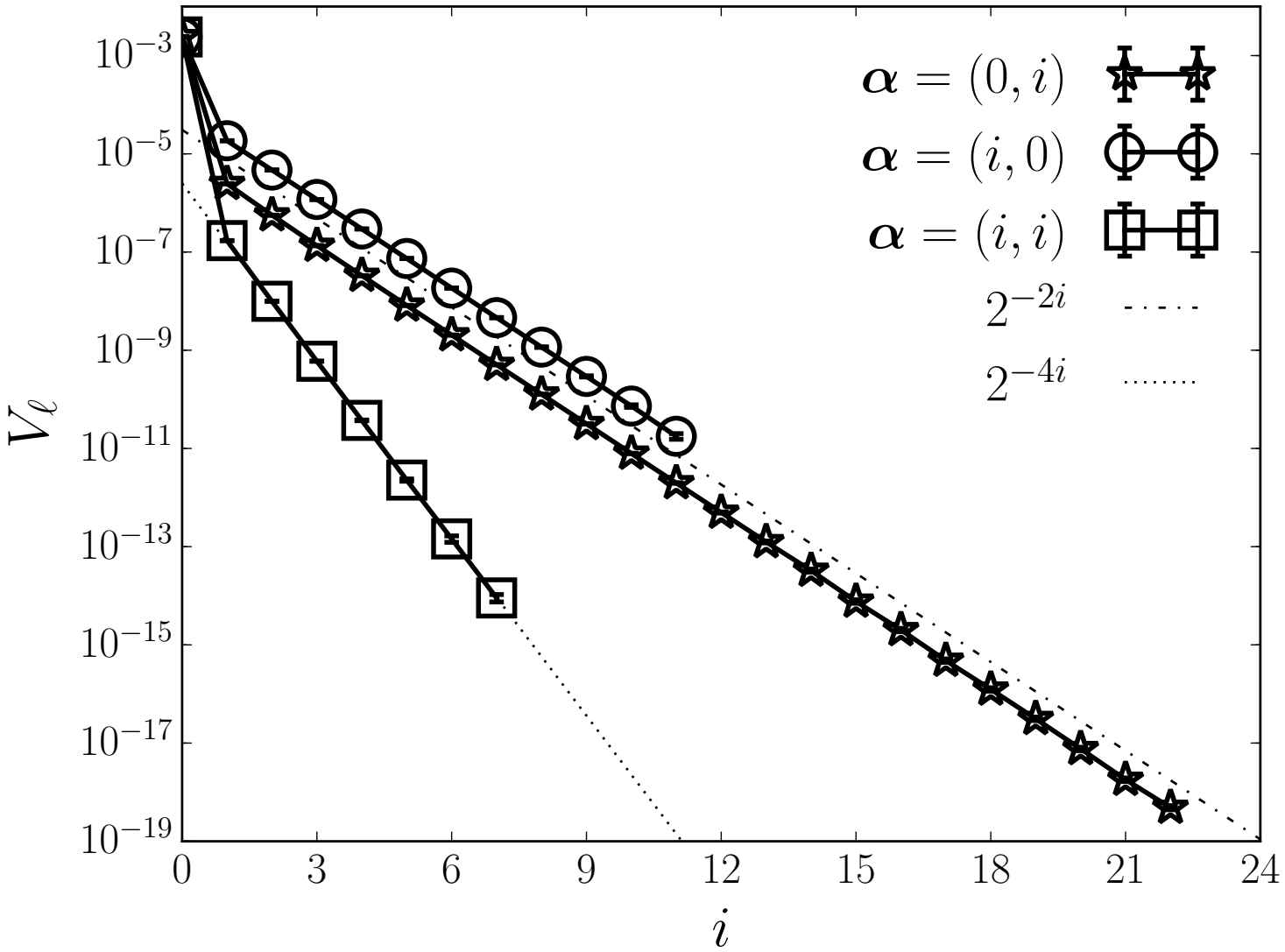
Errors, customised



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